CONTENT RENDERING AND INTERACTION TECHNOLOGIES FOR DIGITAL HERITAGE SYSTEMS

by

Muhammad Zeeshan Patoli

A thesis submitted in fulfilment of
the requirements for the degree of

Doctor of Philosophy

Interactive Systems: Computer Graphics Centre
School of Informatics
University of Sussex
Brighton
BN1 9QJ

December 2010

Declaration

The work described in this thesis, carried out at the University of Sussex is that of the author and has not been submitted in any for any other degree at this or any other university.

Signed ……..…….

Muhammad Zeeshan Patoli

Computer Graphics Centre
School of Informatics
University of Sussex
Brighton
BN1 9QJ

December 2010

Acknowledgements

The work presented in my thesis would not be possible without the help of my DPhil supervisors Dr Martin White and Dr Paul Newbury, I would like to thank both my supervisors whose guidance over the past three years has been highly constructive. I should also thank my fiancée Nazira, all my friends at home, and my colleagues at the Computer Graphics Centre for their support, especially Dr Phil Watten, who have been really helpful during my research.

Additionally, I would like to thank the University of Sindh, Pakistan for providing me financial and moral support in pursuing my DPhil studies here at the University of Sussex.

Finally, and most importantly I would like to express my gratitude to my loving parents, my father, Muhammad Daud Patoli, my mother Zakira Patoli, and all family members for their strong support and understanding. Their emotional support and encouragement during these years strengthened me to complete my degree.

Content Rendering and Interaction Technologies for Digital Heritage Systems

Muhammad Zeeshan Patoli

ABSTRACT

Existing digital heritage systems accommodate a huge amount of digital repository information; however their content rendering and interaction components generally lack the more interesting functionality that allows better interaction with heritage contents. Many digital heritage libraries are simply collections of 2D images with associated metadata and textual content, i.e. little more than museum catalogues presented online. However, over the last few years, largely as a result of EU framework projects, some 3D representation of digital heritage objects are beginning to appear in a digital library context. In the cultural heritage domain, where researchers and museum visitors like to observe cultural objects as closely as possible and to feel their existence and use in the past, giving the user only 2D images along with textual descriptions significantly limits interaction and hence understanding of their heritage.

The availability of powerful content rendering technologies, such as 3D authoring tools to create 3D objects and heritage scenes, grid tools for rendering complex 3D scenes, gaming engines to display 3D interactively, and recent advances in motion capture technologies for embodied immersion, allow the development of unique solutions for enhancing user experience and interaction with digital heritage resources and objects giving a higher level of understanding and greater benefit to the community.

This thesis describes DISPLAYS (Digital Library Services for Playing with Shared Heritage Resources), which is a novel conceptual framework where five unique services are proposed for digital content: creation, archival, exposition, presentation and interaction services. These services or tools are designed to allow the heritage community to create, interpret, use and explore digital heritage resources organised as an online exhibition (or virtual museum). This thesis presents innovative solutions for two of these services or tools: content creation where a cost effective render grid is proposed; and an interaction service, where a heritage scenario is presented online using a real-time motion capture and digital puppeteer solution for the user to explore through embodied immersive interaction their digital heritage.

List of Publications

Conference Papers

N. Collins, C. Kiefer, **M. Z. Patoli**, M. White, "Musical Exoskeletons: Experiments with a Motion Capture Suit", Proceedings of the 2010 Conference on New Interfaces for Musical Expression (NIME 2010), Sydney, Australia, Australia.

M. Z. Patoli, M. Gkion and M. White, "Real-time Online Digital Avatar with Lip Syncing and Facial Expression", ECRT (Early Carrier Researcher's Track) Proceeding of 3rd IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL 2010), held in Kaohsiung, Taiwan, 12th - 16th Feb 2010, ISBN: 978-986-02-2832-8.

M. Z. Patoli, M. Gkion, P. Newbury and M. White, "Real Time Online Motion Capture for Entertainment Applications", The 3rd IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL 2010), held in Kauhsiun Taiwan, 12th - 16th Feb 2010, published in IEEE Computer Society ISBN:978-0-7695-3993-5

W.Zhang, **M. Z. Patoli,** M. Gkion, Al-Barakati, P. Newbury and M. White, "Reanimating Cultural Heritage through Service Orientation, Workflows, Social Networking and Mashups", International Conference on CYBERWORLDS 2009, published in IEEE Computer Society, ISBN: 978-0-7695-3791-7/09, DOI: 10.1109/CW.2009.45

M. Gkion, **M. Z. Patoli,** Al-Barakati, W. Zhang, P. Newbury and M. White, "Collaborative 3D Digital Content Creation Exploiting a Grid Network", "Third International Conference on Information & Communication Technologies ICICT2009" the proceedings are published in IEEE explorer, ISBN: 978-1-4244-4609-4/09.

A. Al-Barakati, W. Zhang, **M. Z. Patoli**, M. Gkion, P. Newbury and M. White, "An Integrated Workflow Management Solution for Heritage Information Mashups" ASONAM 2009, First International Symposium on Mining Social Networks, held in Athens, Greece between 20 - 22 July 2009, the proceedings are published in IEEE explorer, ISBN: 978-0-7695-3689-7/09, DOI 10.1109/ASONAM.2009.40

M. Z. Patoli, M. Gkion, A. Al-Barakati, W. Zhang, P. Newbury and M. White, " An Open Source Grid Based Render Farm for Blender 3D ", presented in IEEE Power Systems Conference & Exhibition (PSCE) 2009, held in Seattle Washington, USA between 15th - 18th March 2009, published in IEEE, ISBN: 978-1-4244-3811-2/09.

A.Al-Barakati, **M.Z.Patoli**, M.Gkion, W.Zhang, P. Newbury, N. Beloff, and M.White, "A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning", Proceeding of 14th International Conference on Virtual Systems and Multimedia (VSMM 2008) dedicated to Digital Heritage, 20 - 26th Oct 2008.

Zeeshan Patoli, Michael Gkion, Abdullah Al-Barakati, Wei Zhang, Paul Newbury, Martin White, "How to Build an Open Source Render Farm based on Desktop Grid Computing", Presented in International Multi-topic Conference IMTIC, Published in CCIS-20, pp. 268-278, 2008.© Springer-Verlag Berlin Heidelberg 2008. ISBN: 978-3-540-89852-8.

M.Z.Patoli, A.Al-Barakati, M.Gkion, W.Zhang, P. Newbury, N. Beloff, and M.White, "A Service-Orientation Approach for a Digital Library System focused on Portable Antiquities and Shared Heritage", Vast 2007, 26 to 29 November 2007, submitted to The 8th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (2007), ISBN 978-963-8046-89-5 (P 15 - 18).

Workshop Papers

M.Z.Patoli, M.White, M.Gkion, W.Zhang, A.Al-Barakati, "Touching the Untouchables", presented in workshop on Touching the Untouchables at University Exeter, UK, http://www.exeter.ac.uk/scienceheritage/MartinWhite.html date: 29 - 30th May 2009

Poster Presentations

M.Z.Patoli, M.White, J.Carol, "Digital Media - 3D Interaction: Personal Motion Capture, Virtual Reconstructions, Tangible Virtual Artefacts", presented in LTN (London Technology Network), London, United Kingdom www.LTNetwork.org

[3] M. Gkion, W. Zhang, A. Al-Barakati, **M.Z. Patoli**, P. Newbury, M. White, *'Mashing up' Digital Worlds for Collective and Exploratory Learning*, CAL09 - Learning in Digital Worlds, Brighton, UK, March 23 - 25th 2009.

A.Al-Barakati, **M.Z.Patoli**, M.Gkion, W.Zhang, P. Newbury, N. Beloff, and M.White, Poster presentation on "A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning", presented in VSMM 2008 - Virtual Systems for MultiMedia dedicated to Digital Heritage, date: 24th October 2008. "http://www.vsmm2008.org/"

M.Z.Patoli, A.Al-Barakati, M.Gkion, W.Zhang, P. Newbury, N. Beloff, and M.White, Poster presentation on "Digital Library Services for Portable Antiquity and Shared Heritage", Informatics Open day, University of Sussex, 29th January 2007.

Glossary

2D	Two Dimensions
3D	Three Dimensions
4DTravel	4D culTural heRitAge Visual experiences through motion capturE and hoLoscopic displays
ACMA	ARCO Content Management Application
AHRC	Arts and Human Research Council
ARCO	Augment Reality of Cultural Objects
ARIF	Augmented Reality Interface
BOINC	Berkeley Open Infrastructure for Network Computing
BRICKS	Building Resources for Integrated Cultural Knowledge Services
BRIDGES	Biometrics Research Informatics Delivered by Grid Enabled Services
BURP	Big Ugly Rendering Project
Codine	Computing in Distributed Networked Environments
CPU	Central Processing Unit
CRUD	Create Retrieve Update and Delete
DCA	Digital Content Archival
DCC	Digital Content Creation
DCE	Digital Content Exposition
DCI	Digital Content Interaction
DCP	Digital Content Presentation
DCP	Digital Content Presentation
DELOS	A Network of Excellence on Digital Libraries
DILIGENT	Digital Library Infrastructure on Grid Enabled Technology
DISPLAYS	Digital Library Services for playing with Shared Heritage
EGEE	Enabling Grid for E-scienceE
EPOCH	Excellence in Processing Cultural Heritage
EPSRC	Engineering and Physical Science Research Council
ERATO	identification Evaluation and Revival of the Acoustical heritage of ancient Theatres and Odea
ESRC	Economic and Social Research Council
EU	European Union
FTP	File Transfer Protocol
GANGA	Gaudi /Athena and Grid Alliance
GIG	Grid Infrastructure Group
GPDK	Grid Portal Development Kit
GPS	Global Positioning System
GRAM	Globus Resource Allocation and Management

GRIDPP	Grid for Particle Physics
HPC	High Performance Computing
HTTP	Hypertext Transfer Protocol
ICT	Information Communication Technology
LHC	Large Hadron Collider
LOCKSS	Lots Of Copies to Keep Stuff Safe
Mocap	Motion Capture
NGS	National Grid Services
OGSA	Open Grid Services Architecture
OGSA-DAI	Open Grid Services Architecture Data Access and Integration
P-GRADE	Parallel Grid Run-time and Application Development Environment
RCH	Reanimating Cultural heritage
SaaS	Software as a service
SDK	Software Development (/Developer) Kit
SDSC	San Diego Super Computing
SGE	Sun Grid Engine
SOA	Service oriented architecture
SRB	Storage Resource Broker
TCP	Transmission Control Protocol
UNICORE	Uniform Interface to Computing Resources
UNICORE	Uniform Interface to Computing Resources
UWCS	University of Wisconsin computer Science (confirm it)
VIRTEX	Virtual Exhibitions
VRML	Virtual Reality Modelling Language
XVRML	XML Virtual Reality Modelling Language

Table of Contents

1 Introduction 1
1.1 Digital Heritage Systems 1
1.2 Problem Statement 2
1.3 Research Questions 5
1.3.1 Scenario: 3D Reconstruction of the Santa Chiara Church 5
1.3.2 Scenario: 4DTRAVEL 6
1.4 Contribution to Knowledge 7
1.5 Proposed Solutions 7
1.5.1 Grid based Content Rendering 7
1.5.2 Interaction through Motion Capture 8
1.6 Organization of the Thesis 8
2 Content Rendering and Interaction 12
2.1 Technologies for Digital Heritage 12
2.2 Service Oriented Architectures 13
2.3 High Performance Computing 16
2.3.1 Tightly-coupled Multi-processors Systems 17
2.3.2 Loosely-coupled Multi-processor Systems 17
2.4 Grid Computing 18
2.4.1 Grid Computing Applications 19
2.4.2 Types of Grid Computing 20
2.4.3 Hardware Components of Grid Computing 23
2.4.4 The Grid Computing Resources 25
2.4.5 The Grid Architecture 28
2.4.6 Grid Middleware Technologies 30
2.4.7 Grid Job Schedulers 34
2.4.8 Grid Resource Allocator 36
2.4.9 Grid Monitoring 36
2.4.10 Grid Portals 36
2.5 Some Popular Grid Projects 37
2.5.1 EGEE 37
2.5.2 TeraGrid 38
2.5.3 NGS 38
2.5.4 GridPP 39
2.5.5 WLCG 39
2.5.6 BRIDGES 39
2.5.7 Climate*prediction*.net 39
2.5.8 SETI@home 40
2.5.9 Monash SPONGE Facility 40
2.6 Motion Capture Systems 40
2.6.1 Types of Motion Capture 43
2.6.2 Motion Capture Industries 49
2.7 Lip Syncing and Facial Animations 50
2.7.1 MotionBuilder 51
2.7.2 Blender 3D 51

2.7.3 Magpie 51
2.7.4 iClone 52
2.7.5 Crazy Talk 52
2.7.6 Papagayo 52
2.7.7 Source Game Engine 52
2.7.8 FaceAPI 53
2.7.9 Annosoft SDKs 54
2.8 Game Engines 55
2.9 Summary 57
3 DISPLAYS Framework 59
3.1 DISPLAYS 69
3.2 DISPLAYS Services 73
3.2.1 DCC Services 74
3.2.2 DCA Services 78
3.2.3 DCE Services 81
3.2.4 DCP Services 85
3.2.5 DCI Services 88
3.3 DISPLAYS Scenarios 91
3.3.1 A Scenario for Portable Antiquity 91
3.3.2 Collaboration in Content Creation 92
3.3.3 Digital Heritage Information Sharing 93
3.3.4 Creating a Virtual Exhibition 94
3.3.5 Real-time Motion Capture based Historical Character 95
3.4 A DISPLAYS System Architecture 95
3.4.1 Service Orientation 96
3.4.2 Workflow Management 99
3.4.3 Summary 100
4 Grid based Content Rendering 102
4.1 Introduction 102
4.2 Render grid for DISPLAYS 105
4.3 Related Work 107
4.3.1 DrQueue 107
4.3.2 FarmerJoe 108
4.3.3 Loki Render 108
4.3.4 BURP-BOINC 108
4.3.5 Deadline 109
4.3.6 ResPower and Rendercore 109
4.3.7 Autodesk Backburner Solution 110
4.4 Render Grid System Architecture 110
4.5 Render Grid Implementation Details 114
4.5.1 Grid Computing Environment 114
4.5.2 Render grid Implementation (Core Functionalities) 118
4.6 Unique Features of this Solution 124
4.7 Test Results and Discussions 127
4.7.1 Test 1: Grid of Six Computers versus Single Computer with Same Specification 128
4.7.2 Test 3: Single Quad Core Machine without Grid Computing vs. with Grid Computing Setup 131

4.7.3 Test 2: Single Corei7 Machine without Grid Computing vs. with Grid Computing Setup 135
4.7.4 Test 4: Render Grid of two Machines (Corei7 and Quad Core) versus Non-Render Grid Environment 139
4.7.5 Conclusion 143
4.8 Summary 144
5 Interaction through Motion Capture 146
5.1 Introduction 146
5.1.1 Digital Puppetry 148
5.1.2 Related Work 150
5.1.3 Limitations of Existing Heritage Projects 150
5.2 Motion capture for Digital Puppetry 152
5.2.1 eMove – Personal Motion Sensing System 154
5.3 Interaction through Motion Capture Heritage Scenario 156
5.3.1 Heritage Role Play Show 156
5.3.2 Digital Puppetry 157
5.3.3 Audience Interacting with the Virtual World 159
5.4 Specification and Design of the Digital Puppet System 159
5.4.1 Digital Puppet Task Execution Sequence 159
5.4.2 Use cases 161
5.4.3 Digital Puppet Architecture 163
5.5 Implementation of the Digital Puppet System 166
5.5.1 Digital Heritage Puppet Application 166
5.5.2 Hardware resources 168
5.5.3 Software resources 171
5.5.4 Network Architecture 183
5.6 Testing and Evaluation 185
5.7 Summary 189
6 Conclusion and Future Work 190
6.1 Cost Effective Render Grid Solution 192
6.1.1 Implementation of Grid Computing 192
6.1.2 Render grid solution based on Grid computing 193
6.2 Real-time Digital Puppetry solution digital heritage 193
6.2.1 Motion capture based visualization using digital puppetry 194
6.2.2 Motion capture based interaction 194
6.2.3 Audio and video technologies for digital heritage 195
6.3 Summary 195
6.4 Future work 196
7 *References* 198
Appendix – A 205
Use Case for Digital Puppetry 205
Appendix – B 215
Setting up Render Grid Environment 215
Appendix – C 218
Setting up Digital Puppetry 218
Appendix – D 221
Snapshots 221

Table of Figures

Figure 2.1: A basic and generic Service Oriented Architecture 15
Figure 2.2: Service Consumer and Service Provider after service binding process 16
Figure 2.3: Grid diagram showing three major Grid Components 24
Figure 2.4: Demonstration of computational resources in a grid environment [23].....26
Figure 2.5: Grid Layered Architecture.....28
Figure 2.6: Grid layers infrastructure and architecture [29] 29
Figure 2.7: Image from 'Avatar' a 3D movie released in 2010. 41
Figure 2.8: Passive Motion Capture System:.....45
Figure 2.9: An actor wearing active markers on different body segments 46
Figure 2.10: An example of magnetic motion capture 47
Figure 2.11: Full body gypsy 6 exoskeleton suit from Animazoo, image source [89].....48
Figure 2.12: On right a female wearing Animazoo IGS-190 suit, on the left side a female wearing similar full body motion capture suit from XSensAnimazoo's IGS -190 [86] XSens full body mocap suit [93]..... 50
Figure 3.1: DISPLAYS services framework 60
Figure 3.2: ARCO system showing various functionalities of ARCO [4] 68
Figure 3.3: DISPLAYS services diagram with tools and services connected to demonstrate the possible workflows.....74
Figure 3.4: DISPLAYS Services diagram with expended DCC service 75
Figure 3.5: Collaborative 3D DCC exploiting a render grid.....76
Figure 3.6: DISPLAYS Services diagram with highlighted DCA service 80
Figure 3.7: DISPLAYS Services diagram with highlighted DCE service.....84
Figure 3.8: DISPLAYS Services diagram with highlighted DCP service.....86
Figure 3.9: Reanimating Cultural Heritage digital repository presentation.....87
Figure 3.10: Re-animating Cultural heritage community on Facebook.....88
Figure 3.11: DISPLAYS Services diagram with highlighted DCP service.....89
Figure 3.12: DISPLAYS Modified Service Oriented Architecture for Reanimating Cultural Heritage 97
Figure 3.13: DISPLAYS Service Oriented Architecture with additional interaction layer 98
Figure 4.1: DISPLAYS architecture deploying the render grid.....106
Figure 4.2: Render Grid Use case diagram 112
Figure 4.3: DISPLAYS render grid architecture for open source Blender3D 113
Figure 4.4: The machines used in Render grid setup (one master, six slaves and a router) 117
Figure 4.5: Interface to the render grid system management environment 119
Figure 4.6: Chop Animation also named as Script Generator dialogue box 121
Figure 4.7: Job submission script, for grid environment to render animation file.....122
Figure 4.8: Render grid job monitoring tool showing a job in progress on six render machines.....123
Figure 4.9 One of the four hundred rendered frames from Fishbourne Roman Palace.....129
Figure 4.10: Rendering results comparison of grid of seven computers with single computer 129

Figure 4.11: Comparison of render grid versus none-render grid solution for small size images on Quad core processor 132
Figure 4.12: Comparison of render grid versus none-render grid solution for medium size images on Quad core processor 133
Figure 4.13: Comparison of render grid versus none-render grid solution for large size images on Quad core processor 135
Figure 4.14. Time consumed by a Corei7 CPU with Grid and without Grid Computer 136
Figure 4.15: Comparison of render grid versus none-render grid solution for medium size images on Corei7 processor 137
Figure 4.16: Comparison of render grid versus none-render grid solution for large size images on Corei7 processor 138
Figure 4.17: Comparison of render grid versus none-render grid solution for small images 140
Figure 4.18: Comparison of render grid versus none-render grid solution for medium size images 141
Figure 4.19: Comparison of render grid versus none-render grid solution for large images 143
Figure 5.1: Lee Harrison III, photo montage featuring a dancer with body mounted sensors controlling real-time animation on the ANIMAC, 1962, Denver [161] 153
Figure 5.2: eMove Personal Motion Capture Suit: on left initial illustrated version of the suit, on right side CAD design of final version of eMove suit. 155
Figure 5.3: DISPLAYS Digital Puppetry and Real-time Heritage Interaction 158
Figure 5.4 Digital Puppetry Sequence Diagram to Initiate Client and Server Communication 160
Figure 5.5: Use Case Diagram, Showing Possible Use Cases for Digital Puppetry 162
Figure 5.6: Digital puppet system architecture showing various components in client server architecture 163
Figure 5.7: digital puppetry for digital heritage: client-server architecture 167
Figure 5.8: Creative Sound Blaster X-Fi Go! 169
Figure 5.9: (a) Author and developer of this work wearing development version of eMove motion capture suit. (b) CAD design for final version of eMove suit to be released by mid-2011 170
Figure 5.10: Display.exe file showing mocap raw channels data 172
Figure 5.11: eMove SDK connected with motion capture data through Display.exe 173
Figure 5.12: The IP selector dialogue box for 'Enable server' option inside eMove SDK 173
Figure 5.13: A character inside MotionBuilder connected with remote machine through eMove SDK (Real-time Motion capture data) 175
Figure 5.14: eMove voice chat call dialogue box, available in 'Chat' menus in eMove SDK 176
Figure 5.15: eMove video chat (server side) window to view remote (client) camera 176
Figure 5.16: Digital Puppetry Client Side View 177
Figure 5.17: Mocap data flow diagram from eMove SDK to game engines 179
Figure 5.18: A graph showing score for test results for animation data between Brighton, UK and other places 187

Figure 5.19: A graph showing score for test results for audio data between Brighton, UK and other places 188
Figure 5.20: A graph showing score for test results for video data between Brighton, UK and other places 188

Table of Figures

Table 3.1: Comparison of several key digital library systems features against DISPLAYS 64
Table 4.1: Comparison between different available render farms 125
Table 4.2: Feature Key: 125
Table 4.3: Rendering details of small images on Quad core processor 132
Table 4.4: Rendering details of medium size images on Quad core processor 133
Table 4.5: Rendering details of large images on Quad core processor 134
Table 4.6: Rendering details of small images on Corei7 processor 136
Table 4.7: Rendering details of Medium size images on Corei7 processor 137
Table 4.8: Rendering details of large images on Corei7 processor 138
Table 4.9: Rendering details and comparison of small images on corei7 and Quad core processor with and without render grid solution 140
Table 4.10: Rendering details and comparison of medium size images on corei7 and Quad core processor with and without render grid solution 141
Table 4.11: Rendering details and comparison of large images on corei7 and Quad core processor with and without render grid solution 142
Table 5.1: shows the Digital puppetry tests results on the scale of Worst to Excellent 186

CHAPTER I

1 Introduction

1.1 Digital Heritage Systems

Preserving the past for the future has been the slogan for scientific and digital heritage libraries. In other words the preservation of valuable, extremely fragile, inherently impermanent, and difficult to access heritage resources is the main objective of any scientific and digital heritage system. The functionalities of a common digital heritage system are the creation of digital objects, archiving them for long term sustainability, providing easy access for users and a system management environment, also called a system interface for managing heritage contents.

Generally digital heritage libraries are a set of services used to manage digital representations of heritage objects. Basic services should be provided that allow objects to be created, retrieved, updated and deleted (CRUD) by authorized users through a networked-based system; other operations may also be possible, for example the ability to build expositions of objects, present them and interact with them. This allows users from various geographical locations to access a huge knowledge base without travelling to the location where these heritage objects physically exist; this also allows users to create multiple digital copies of these objects wherever required to present these objects anywhere without fearing of losing the actual object. The archival of such objects in digital format is useful in the sense that physical objects are often unavailable, i.e. stored in physical archives, and even though physical objects are preserved through

advanced methods, a regular museum visitor cannot be given open access and interaction with these objects due their fragility and preciousness.

In the past many digital heritage projects have been funded in Europe by national funding bodies such as the EPSRC (Engineering and Physical Sciences Research Council), AHRC (Arts and Human research Council) and ESRC (Economic and Social Research Council), and by the EU (European Union). Projects such as DELOS [1], DILIGENT [2], BRICKS [3], ARCO [4], Reanimating Cultural Heritage [5] are more notable examples, and they are briefly discussed in Chapter 3.

The work in this thesis is based around DISPLAYS (Digital Library Services for Playing with Shared Resources) [6], [7], which is a conceptual framework (used in the context of this thesis to establish requirements for DISPLAYS components, of which many combinations can exist to implement a DISPLAYS architecture) utilizing different technologies and methodologies to produce innovative and useful digital library services for the cultural heritage community. In the context of the thesis actual DISPLAYS implementations are composed of experiments, prototypes, individual results, tests and proofs of the overall concept by the DISPLAYS research team. Two complete implementations can be regarded as the ARCO system (which is subsumed by DISPLAYS) and the ‘reanimating cultural heritage’ system currently being completed. Chapter 3 describes the DISPLAYS concept, its overall architecture embodied with new technologies, concepts and other proposed innovations for the cultural heritage discipline. This thesis is concerned with the implementation of two innovative components (services or tools) for the DISPLAYS concept.

1.2 Problem Statement

Existing digital library infrastructures are able to accommodate huge amounts of digital repository information. However when compared to similar ICT knowledge bases in other domains, these digital libraries, their visualization and interaction parts are lacking in many advanced functionalities, notably in the creation of 3D content and interaction methods for that content. Development of new innovations in this area can make museums and archaeological sites a centre of attraction and knowledge for their

collections, while at the same time stimulate a greater interest in their collections. For example, one can ask the following questions about existing digital heritage systems:

1. Why should digital heritage libraries or systems support 3D graphics and animation to portray their heritage content?
2. Are these digital heritage systems capable of processing complex 3D images and animations effectively? Do they have enough computation power or techniques to render these 3D animations for advanced digital heritage content?
3. Do existing digital heritage systems have enough potential to meet end-user requirements for visualization and interaction compared to the fast growth in other similar ICT technologies?
4. Is it easy to understand the objects, their environments, history and background by reading the text and looking at static images of objects or even physical objects?
5. Is there any well-defined and easy to understand relationship between the objects, their environments and their owners or users?
6. Can these digital libraries survive much longer with only 2D images and movie clips in the current era of virtual and augmented reality worlds, gaming engines and new interaction techniques?

The general answer to these questions is often in the negative. Why?

1. In the cultural heritage domain users like to touch and observe real life objects to see, feel and sense these heritage objects, and view these objects from different angles to learn more about them. Giving them only two 2D images of these objects is a poor substitute. There is the need to introduce different virtual and augmented reality methods to retain the user's attention towards the museums and associated heritage websites.
2. Museums usually have limited budgets for the conservation of heritage objects but no budget for high performance computing resources. However, a large national museum will have perhaps many hundreds of PCs lying idle during breaks from general keyboard use. A museum investing in 3D to create a

complex 3D heritage scenario could have access to useful resources in-house within a DISPLAYS concept.

3. There is nothing attractive from a visualization and interaction point of view when comparing current heritage content (image and text) with virtual and augmented reality based content, particularly when accessed through innovative interfaces such as gaming engines. Most existing digital heritage is either a collection of 2D images, videos or a combination of both with the exception of some with only a little 3D. Usually these images or gallery of images or videos are controlled by either keyboard or mouse or they are just play back videos (sequential images). There is nothing exciting for the end-users in 2D images or video as these are all predefined and there is no real user interaction (i.e. users cannot see any new perspective of objects or environments by themselves).
4. Existing digital heritage projects are usually web-based and either a collection of 2D images or 3D objects only, with some written descriptions. These objects do not present their history unless the visitors read their text properly and understand it from the curator's view point. Research shows the visual aspect of any scenario draws more attention and makes more sense for the observer. Robert Grudin, who received a PhD in Comparative Literature, wrote many books about the Time and the Art of Livings, etc. He said:

   > "Written truth is four-dimensional. If we consult it at the wrong time, or read it at the wrong pace, it is as empty and shapeless as a dress on a hook" [8]

   Hence a visual scenario showing all objects laid inside their original environment, along with their original usage and user, can explain the whole story by itself in a user (museum visitor) understandable way is what is required for cultural and digital heritage museums.
5. The relationship between objects and their environment appears as a textual description alongside the objects in museums. These textual descriptions are summarized and kept with the objects inside the museums and are explained in more detail in textbooks, often kept in a separate library. There is a need to

combine objects, their respective environments and stories in a visual format rather than textual format – a so called visual narrative.

6. Most digital library archives are located in one place; a museum curator archives newly arrived objects, and then these digital objects are made available to the public through web pages and search engines. There is an immense need for new technologies such as 3D animations (using game engines), interaction using appropriate user controllable interfaces, and, most of all, the static text from curator's comments to textual histories need to be brought alive (in a visual and interactive format) to better explain the history (e.g. story of an object).

Given these particular problems, this thesis sets out to investigate a limited set of research questions focused on a set of services or tools that target content rendering and interaction within a digital heritage system concept (DISPLAYS).

1.3 Research Questions

Two scenarios are suggested here that help to formulate specific research questions that drive the work completed in this thesis – there are many others discussed in Chapter 3, 4 and 5 of this thesis.

1.3.1 Scenario: 3D Reconstruction of the Santa Chiara Church

The Victoria and Albert Museum require a 3D reconstruction of the Santa Chiara Church, Chapel and Altar. The medieval renaissance church located in Florence, Italy was deconsecrated in 1808 and in modern times has been converted to office and domestic premises. The Chapel and its three altars (two side and the high altar) were removed and sold to the Victoria and Albert Museum in 1861; they are now located in the new medieval and renaissance galleries. A clear need arises to digitally repatriate the church, chapel and altars for the benefit of Florence and its people. The Victoria and Albert museum, on a limited budget, would like to receive their 3D reconstruction as part of photo-realistic walkthrough, i.e. a 3D movie lasting about 3.5 minutes. The Victoria and Albert Museum approach the DISPLAYS team because they know they have a cost effective grid rendering technology (which will be required to render the

frames for the movie) that can utilise the existing Victoria and Albert Museum computing resources, and hence cut the cost of the commission.

The question then arises: can an alternative to a render farm based on grid technologies (i.e. a render grid) be developed that is cost effective and that could be deployed over an 'institution's' (large or small scale museum for example) computing resource such that it could be used in the above scenario?

1.3.2 Scenario: 4DTRAVEL

A local archaeological museum has a collection of artefacts from a local archaeological site. The museum wishes to 'bring to life' their collection through a re-enactment method where characters of the period narrate a story about the site to a group of visitors. They have considered several methods, including employing a physical actor to dress in costume of the period to narrate the story of the site; however they have heard of a possibility of replacing the actor with a 'virtual character' who can narrate the story, which also allows the story of the site to be told online.

The question then arises: How is it possible for a virtual character to provide a more engaging interaction with digital heritage that can facilitate users to understand their heritage more effectively? How can digital heritage systems and their archived objects be made more interesting and useable for a heritage lover? What could be done to make cultural heritage more interesting, i.e. how can we make museums and archaeological sites a point of attraction for everyone rather than heritage lovers only? Also, what kind of innovative technologies are required to accomplish this task?

In summary, there is a need for introducing new techniques and technologies to digital heritage sites, monuments and objects to retain everyone's interest in heritage resources. This should also preserve our cultural heritage by giving them more interesting ways to play with virtual heritage objects, eliminating the need for touching the actual physical heritage objects.

1.4 Contribution to Knowledge

- A novel and cost effective render grid implementation using wasted CPU power that allows institutions to employ their own resources for cost effective rendering of complex 3D reconstructions and simulations of digital heritage scenarios.
- A real-time digital puppetry (virtual character) solution to enhance digital heritage engagement that can easily be controlled by a novice operating an historical avatar inside a heritage environment.
- Design and implementation of an associated audio, video and text interaction facility that can be used inside a digital heritage world for online engagement between the historical avatar and a museum visitor.
- Contribution to the DISPLAYS conceptual framework by validating and testing the results from the above contributions.

1.5 Proposed Solutions

This thesis presents two solutions to the problems outlined above:

1. ***Heritage content rendering***, where there is a need for a cost effective render grid solution that can be helpful for digital artists working on digital heritage content creation or heritage modeling.

2. ***Development of novel techniques for heritage interaction,*** which can make digital heritage resources as a key point of attraction for heritage community members as well as common users.

1.5.1 Grid based Content Rendering

A cost effective render farm type solution (i.e. a render grid) has been proposed as a part of this research. The solution involves setting up a render grid using old or existing machine power rather than buying an expensive blade or dual processor server machines, or renting an expensive render farm. This kind of solution is useful for users who have several older machines and they want to use these machines as a render farm

to render their creations, and also in the case where more than one artist or group members are working on a similar or different project but they are close to each other. They can then use available machine power as a render grid when these machines are free (e.g. during night time, or lunch and coffee breaks, or even when machines are idle). Indeed, for larger national museums with many PCs a very powerful rendering solution can be created for generating 3D reconstruction fly-through movies of ancient sites and monuments.

1.5.2 Interaction through Motion Capture

This research introduces some new methods, techniques and technologies, which have never been introduced to digital and cultural heritage before. The new proposed technologies, i.e. motion capture systems along with 3D animation, video and audio features are investigated for application to digital heritage interaction scenarios. These technologies, together with methods and scenarios, will attract users who want to deepen their understanding of their heritage past to actually come to museums or heritage sites to see, feel and interact with the heritage environment and objects.

This methodology includes: building a digital puppetry (heritage character) application for the visualization of digital heritage objects and digital heritage role playing games, where characters from the past are brought to life.

1.6 Organization of the Thesis

This thesis is organized so as to present two core research works, namely 1) a render grid solution for digital heritage environments; 2) motion capture, audio and video technologies for digital heritage visualization and interaction. The thesis as a whole presents a single body of research, with each chapter containing an introduction and a summary that justifies its existence within the thesis.

Chapter 1 sets the scene, i.e. an introduction to the work carried out in this thesis. This chapter alludes to existing flaws and benefits in digital heritage library infrastructures and the need for new techniques and technologies, in particular relating to content rendering and interaction. A problem statement has been derived along with their

reasoning and, on the basis of this problem statement, a set of scenarios and questions that led to this research has been presented. In addition, a set of proposed solutions in the light of cutting edge technologies have been discussed.

The next chapter, Chapter 2, is a detailed technological background and critical literature review of the key research areas related to the work that was outlined in Chapter 1. This literature review chapter sets the scene for the technical background to this thesis by covering background technologies, scientific concepts and related works. The chapter specifically covers existing digital heritage infrastructures, service oriented architecture, grid computing and render grid, real-time motion capture technologies, lip-synchronization and facial expressions, 3D animation environments, audio and video features. The purpose of this chapter is to introduce these technologies and justify their requirements inside digital heritage systems.

Chapter 3 is a detailed description of the DISPLAYS framework, which presents the requirements and specifications proposed for scientific and technological updates in digital heritage libraries. The chapter introduces the DISPLAYS framework with its five proposed services, a set of scenarios where these services are applicable, its system architecture, services orientation, and system workflow management, followed by concise conclusion.

Chapter 4 presents core level design and implementation details about the innovative render grid solution proposed in this thesis. In this chapter, a detailed description of the render grid solution, its comparisons with other available options, benefits and flaws are discussed. A high level system design is drawn and explained for its software and hardware level architecture. This chapter also includes methods of implementing a grid computing infrastructure using available hardware and software resources and implementation details of a rendering solution on the top of a grid computing infrastructure. Furthermore, the seven core functionalities, Job Registry, Chop Animation, Job Submission, Render Grid, Job monitoring, Retrieve Rendered Results, and Encode Results, are explained. Finally, this chapter concludes with discussions on the testing and evaluation phase of the implemented render grid solution, followed by a

discussion that compares the capabilities of the render grid against a single computer of the same specification.

Chapter 5 contains the second part of the design and implementation phase, which is about introducing innovative motion capture technologies (real-time historical character animation). Here motion capture technology, coupled with audio and video features, is proposed for digital heritage systems to retain the user's interest in digital heritage systems through innovative methods of interaction with digital heritage objects, environments and avatars. This chapter describes the concepts of digital puppetry, heritage role playing scenarios, their possible roles inside digital heritage environments, some relevant work found in literature, and key differences between this work and existing projects in the same area.

This is followed by system design and implementation principles for digital puppetry applications and its sub-components; these include eMove-chat, eMove SDK, etc. These components, eMove-chat and eMove SDK, are part of a real-time digital puppetry and online gaming application that the author of this thesis has designed and adapted for digital heritage applications; however these same components find even greater use in other digital entertainment applications, such as theme parks where, for example, a real-time digital puppet or virtual character can be used to entertain theme park visitors. In the context of this thesis, the concept of 'digital heritage puppetry' or 'historical character animation' for 'embodied interaction' (embodied interaction is where a real person can take on the persona of a virtual character immersed in the digital heritage world), is demonstrated through two working prototype 'games' that illustrate what could be done in a chosen heritage scenario. The chapter concludes with a concise description of results drawn from various tests performed on an online (remote) version of a digital puppetry application for digital heritage systems and followed by its conclusion.

Chapter 6 concludes the research work carried out in this thesis with an outline of implementation, test and results. Furthermore, this chapter also highlights the future intended work in both research directions (i.e. render grid and digital puppet application

using motion capture technologies) on the basis of successful test results achieved during this research work.

CHAPTER II

2 Content Rendering and Interaction

2.1 Technologies for Digital Heritage

This thesis introduces new technologies for digital heritage systems for interaction and rendering environments proposed as part of the DISPLAYS framework. These technologies have not been used previously for digital heritage systems; however this thesis proposes that using them inside heritage environments will provide remarkable changes in heritage interaction and content creation services. The DISPLAYS architecture and its all functionalities are presented in Chapter 3, whereas this chapter presents the key technologies that are useful for the innovative interaction and rendering solutions developed as part of this work:

- Service Oriented Architecture
- High Performance Computer related to Rendering
- Real-time Motion Capture Technologies
- Lip-Synchronization and Facial Expressions
- Audio and Video Features

Service oriented architecture is proposed as an underlying architecture for DISPLAYS, i.e. where possible all functionalities should be implemented as web services and used by a single workflow management system. The render grid is recommended for providing a cost effective rendering solution to heritage object modellers. Motion capture technologies will provide an innovative way of interacting with heritage objects and navigating through reconstructed virtual heritage sites. Lip-synchronization and

facial expressions are essential for playing a believable role as an avatar inside a reconstructed heritage environment. The audio functionalities are included for communication between avatars inside a heritage world or between an avatar and a real life character over the network; video features are included to see the feedback of an audience and to respond them interactively.

This chapter provides the brief technical background of these technologies, their alternative possibilities and comparisons between them.

2.2 Service Oriented Architectures

The Service Oriented Architecture (SOA) is an architecture that produces a modular design of applications, where every component of the system (called service) can be located in geographically disbursed areas and can be used by any system anywhere in the world on any architecture. DISPLAYS propose the adoption of a service orientation approach as does BRICKS [3], see Chapter 3 for a more detailed discussion. This discussion is limited to an explanation of service orientation in general and how it may be applied to digital heritage systems.

The major benefits of SOA include portability, robustness and reusability of services by different systems, and its distributed architecture allows sharing workload on different machines and collaboration. The beauty of such architecture is that system users have a single interface to all services, and they access the system without knowing about the background of these services. This is even though the actual system architecture may be quite complicated and these services may also be serving many other applications. Different services on different machines allows the system to be portable, and any occurrence of error in part of the system will not cause whole system crash, but only the affected part.

One can find different definitions of SOA given in different perspectives: below are some relevant definitions to describe SOA in information technology and business fields:

> "An SOA is a business concept, an idea or approach, of how IT functionality can be planned, designed, and delivered as modular business services to achieve specific business benefits" [9].

The modular design of SOA architecture allows the business planners to follow the divide and conquer principle to solve large problems and facilitate them with easy troubleshooting in case of any bug or problem in system. The IT industrial giant IBM defines SOA as:

> "Service-Oriented Architecture is an IT architectural style that supports the transformation of your business into a set of linked services, or repeatable business tasks that can be accessed when needed over a network. This may be a local network, it may be the Internet, or it may be geographically and technologically diverse, combining services in New York, London, and Hong Kong as though they were all installed on your local desktop. These services can coalesce to accomplish a specific business task, enabling your business to quickly adapt to changing conditions and requirements". [10]

The SOA is a distributed architecture where individual components (services) of the system can be located in any part of the world, or these services can be part of a different system, which allows these services to access only particular functionalities. Services such as, YouTube, Flickr, Google, Yahoo, Bing, etc., provide web services to share certain elements of their system that becomes part of other systems.

The SOA is essentially a collection of services, where all services are loosely coupled with each other through some sort of mechanism called a service registry or database. All the services register themselves inside the service registry in order to be accessible to other processes or user applications. This registry is basically a database containing information about all the related services and their accessibility information such as addresses/location of service. Any other service or user application, who wishes to gain access to any of the listed services, needs to contact the system registry with its requirements to see if any matching service is available. The registry matches the requested service with registered services and produces the access to a particular service, which can then be accessed by the system.

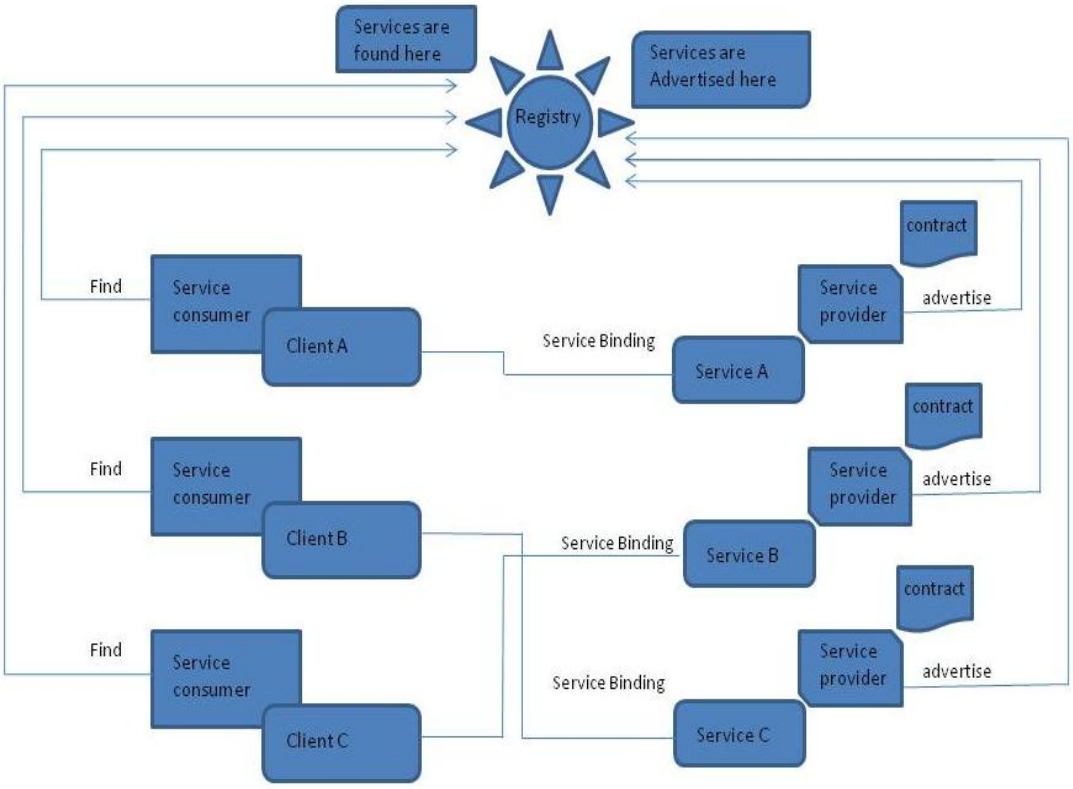

Figure 2.1: A basic and generic Service Oriented Architecture

Figure 2.1 above is a generic service-oriented architecture, displaying a service registry and some sets of services; each service is held by a service provider who advertises the service into a service registry. Listing a service into a registry is based on some sort of a contract, and the service consumer needs to fulfil the contract conditions in order to gain access to the required service. The consumer is a client who needs a particular service: the consumer contacts the service registry with its requirements, and the registry provides the access to a particular service for a limited time based on agreed terms. This process is called service binding.

The services are a base for service-oriented architecture; they need to be designed in such a way that they can assure reusability, interoperability, and integration across all platforms. These services can also communicate with each other through the identity and address provided by a registry service mechanism. The communication can involve either simple data passing or it could involve two or more services coordinating some activity.

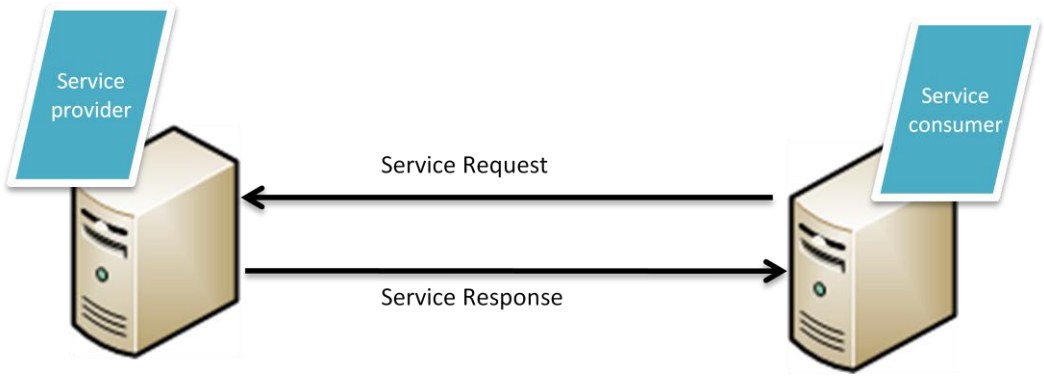

Figure 2.2: Service Consumer and Service Provider after service binding process

Once the service binding process is confirmed through registry, a direct communication channel is established between the service provider and service consumer to access the service. The service consumer requests a service and the provider provides the service based on the request. Figure 2.2 demonstrates this concept, with two machines acting as a service provider and a consumer.

2.3 High Performance Computing

Typically High Performance Computing (HPC) can be defined in a system with the following capabilities:

- Significant computational power
- The need to access and/or process very large amounts of data quickly
- The need to operate interactively across a geographically distributed network

A personal computer would never be sufficient to fulfil the requirements of many scientific projects. The majority of scientific projects either require a multi-processor system (a computer system having more than one microprocessor), a supercomputer (the fastest computer of an area), or sometimes a loosely coupled parallel system (e.g. cluster or grid computing).

The HPC system could be a computer system with one powerful microprocessor, or with multiple microprocessors, or it could be more than one system with single or multiple microprocessors all working to solve a single problem, and working under a team leader that is a again a microprocessor. A single processor is usually not enough to

perform HPC tasks. The more powerful systems usually require more processors called multi-processor systems. There are two kinds of multi-processor system:

2.3.1 Tightly-coupled Multi-processors Systems

The multi-processors are tightly coupled systems where all the micro-processors reside on same motherboard inside the same computer system. They share common buses, memory, input/output and other resources. These kinds of machines are more powerful than a loosely coupled system with an equal number of processors, but they are extremely expensive due to their design architecture.

2.3.2 Loosely-coupled Multi-processor Systems

The loosely coupled multi-processor systems are more than one computer system where every machine has its one memory, buses, I/O devices, etc. These machines are connected through an external cable such as a network cable, and are configured to act as a single computer. The loosely coupled systems are categorized on the basis of whether the machines are dedicated or non-dedicated for HPC infrastructure. Below are two kinds of loosely coupled system with their brief description.

2.3.2.1 *Cluster Computing*

Cluster computing is one of the ways to achieve a high performance computing and supercomputing environment. It is a kind of loosely coupled parallel processing unlike dual processors where all micro-processors are coupled together on the same motherboard. The cluster computers are similar to tightly coupled systems as all processors are dedicated to clusters only. Every cluster has slaves and a master computer; the slave computers orders from master computer. Usually these systems have:

- Single Operating system on master node/master computer
- Slave with no operating system and are usually booted via network
- The cluster can utilize 100% resources (processor power, RAM, etc.) of all nodes.

2.3.2.2 *Grid Computing*

Another way to achieve high performance computing is grid computing. This is different from a dual processor system as all the processors have their own memory, buses and hard drive space. It also differs from cluster computing as the slave machines used as part of a grid are not dedicated to work as a grid unit only, but these machines can also be used for other tasks. In the case of grid computing, a machine is already used by a user, so its 100% CPU time is never used and can be utilized to work as a grid unit.

The main differences between cluster and grid computing are:

- Cluster computing requires dedicated computers to work together as a part of the clusters, but a grid does not necessarily require a dedicated system.
- Cluster computing can always utilize 100% CPU cycles of slave nodes, but in the case of a grid, only wasted CPU cycles can be utilized.
- In many cases clusters of computers are made as part of a grid, to get more powerful computing environment.

2.4 Grid Computing

A grid is a collection of distributed computing resources that can be accessed by the end users of the grid both on wide area network (WAN) and local area network (LAN). These resources appear as a single large virtual computing resource over the network hiding all the background details from its end users.

> "The Grid computing is a kind of distributed computing environment that enables dynamic selection, sharing and aggregation of geographically distributed resources based on availability, capability, performance, and cost of these resources" [11].

The applications developed for grid environments require attention the during development stages, these applications need to be designed as distributed applications that can work on a multi-processor grid infrastructure. Alternatively the grid interface

and job scheduler need to be robust enough to deal with existing non-distributed legacy applications and distribute them between grid nodes.

The grid computing is also named as, “computational grid”, “Just-in-time computing”, “utility computing”, and “high performance distributed computing”.

2.4.1 Grid Computing Applications

There are many applications for grid computing. Ian Foster and Carl Kesselman have identified five major application classes for grid computing in their book “the Grid: Blueprint for a New Computing Infrastructure” [12], also discussed in an invited talk at the VECPAR2000 conference: the proceedings can be found at [13]. Below is short description of these five main categories.

2.4.1.1 *Distributed Supercomputing Applications*

There are many applications that cannot be fulfilled by a single computer; this involves CPU power and computer memory. Hence, the application is distributed between more than one computer system to accomplish the same task. The distributed supercomputing applications uses grid computing to aggregate computational power from various dedicated machines to execute the task that cannot be executed by a single computer.

2.4.1.2 *High Throughput Computing Applications*

In high throughput computing, the applications are distributed on many computers connected by a network, but only those that have free (idle) computational cycles are used. The machines that are already busy in running some other applications are not interrupted. This process of stealing the idle CPU cycles is also called CPU Scavenging.

2.4.1.3 *On-Demand Computing Applications*

On-demand applications use a grid whenever required; this could be for a short term or long term, but when an application is finished the guest leaves the host. This is especially useful when the resources are extremely expensive (e.g. computation, software, data repositories, specialized sensors, etc.), and can be made available

temporarily. Once the application execution is finished the resources are released and made available for other applications

2.4.1.4 *Data Intensive Applications*

The applications that require huge amounts of data to be stored somewhere in a geographically distributed area such as digital libraries, high energy physics and mathematics data calculations results, digital image repositories, etc are covered in this category. The data intensive applications not only require storage resources to archive data, but often these kinds of applications require additional resource such as computation power and network bandwidth to process and transfer this data in a networked environment.

2.4.1.5 *Collaborative Applications*

Collaborative applications require human to human interaction to share experiences and expertise. Often these applications require collaboration for data, information and knowledge sharing. Hence the grid can be used to serve these kinds of applications.

2.4.2 Types of Grid Computing

Grid computing is becoming more popular from academics to highly popular levels (i.e. research projects, country level grids, countries cross boundaries grids, and in industries). Also there have been many new terms introduced for grid computing; some of the popular terms found in literature [14] are:

- Compute/Computational Grids
- Data Grids, Science Grids
- Access Grids
- Knowledge Grids
- Bio Grids
- Sensor Grids
- Cluster Grids
- Campus Grids

- Tera Grids
- Commodity Grids

There are different derivations of grid computing found in research. Discussion on each of them is beyond the scope of this thesis, but there are three main categories (namely Computational Grid, Scavenging Grid and Data Grid) [11], which describe most purposes for building grid environments and classify the existing grid infrastructures:

2.4.2.1 *Computation Grid*

In a computation grid the machines that appear as a part of a grid are assigned a compute intensive task (like: calculations of iterative mathematical expressions, etc...), and all computers resolve their assigned problems and returns the result to the master computer. This kind of grid acts as a computing cluster to solve a particular task during the execution time, but they can be used for other purposes unlike cluster computing.

The founder of Grid Computing, Ian Foster defines a computation grid as:

> "A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities." [15].

The objective in computation grids is to develop a common interface to all computation cycles attached in the grid; hence all computers act as a single computer. This also includes developing dedicated grid-based clusters from different sources. The applications need to be designed in such a way that they are compatible to run on computation grids (i.e. the user needs to develop distributed grid applications or a mechanism to distribute the existing applications).

2.4.2.2 *Scavenging Grids*

Scavenging grids are also called "Desktop Grids". This type of grid is built to produce a low cost grid environment using existing machine power. This applies where the normal home and office machines are not used for 100% of their time, they are usually either idle or only some part of its CPU cycles are normally used, hence most of the time these

CPU remains idle. CPU scavenging is famous for harvesting these idle CPU cycles from local networked systems, or sometimes the Internet. This type of grid normally requires the installation of a small piece of software that tells the master/server when it is idle for some time, and then the master computer starts assigning the jobs to it. Once the original user interrupts the machine any executing task is terminated and machine control is returned to its user.

This is particularly useful for organizations where high performance computing is required, funds are limited but many personal computers are available and are used for small tasks for a limited time. A practical example of such organizations could be a research university or colleges with low or no budget for HPC. The DISPLAYS grid infrastructure is also proposed on the basis of the CPU scavenging model, so that we can benefit the heritage community with low cost modelling and rendering environments by rendering the models on a grid infrastructure.

2.4.2.3 *Data Grid*

The data grid allows an organization to create a unified interface to all data repositories located in a networked environment; this provides a central control to all individual data elements. The user can query the data on distributed repositories located on various places using the interface provided by a grid central manager, and similarly it can input new data into various repositories through the same interface. Hence the user gets a central management over distributed database management systems that may be located in various places around the world. Typically the data on which scientists and researchers work are usually flat files and these files are required on different machines in many countries. The data grid provides a well organized, structured and reliable way to archive the digital form of data, and provides easy access to retrieve them.

The variation of a Data Grid is Knowledge Grid, where the digital objects and data (such as images, 3D models, movie clips, sound clips, astrological data or geographical maps) are combined, stored, structured and presented in such a way that they present the knowledge for the end user [16] [17]. An author of the article “the knowledge grid” has proposed useful three layer architecture for designing, building, and implementing distributed knowledge discovery services for a data grid [18].

There are many ongoing projects on data and knowledge grid architecture and services; a good example is "KnowledgeGRID Malaysia" which provides data storage and computing facility to researchers in Malaysia [19].

2.4.3 Hardware Components of Grid Computing

The grid is a network-based infrastructure that requires many hardware and software components to perform its operations. The software components include network protocols, middleware technologies, security layers, operating systems, and many other application dependent packages. There are three main hardware units/components in grid computing infrastructure: 1) Grid job submitter (user computer), 2) Central manager/grid job scheduler, and 3) Execution servers. All of these could be expensive machines or could be day-to-day normal computers.

Figure 2.3 below shows three important hardware components, in different blocks. Note: It is not compulsory to have three different machines to act as a central manager, job submitter and execution servers. One machine can perform any two or in fact all three options for testing purposes, but a real grid computing infrastructure must have separate execution servers and a central manager. It is normal practice that many users install central manager and job submitter components on same computer along with many other remote job submitter machines.

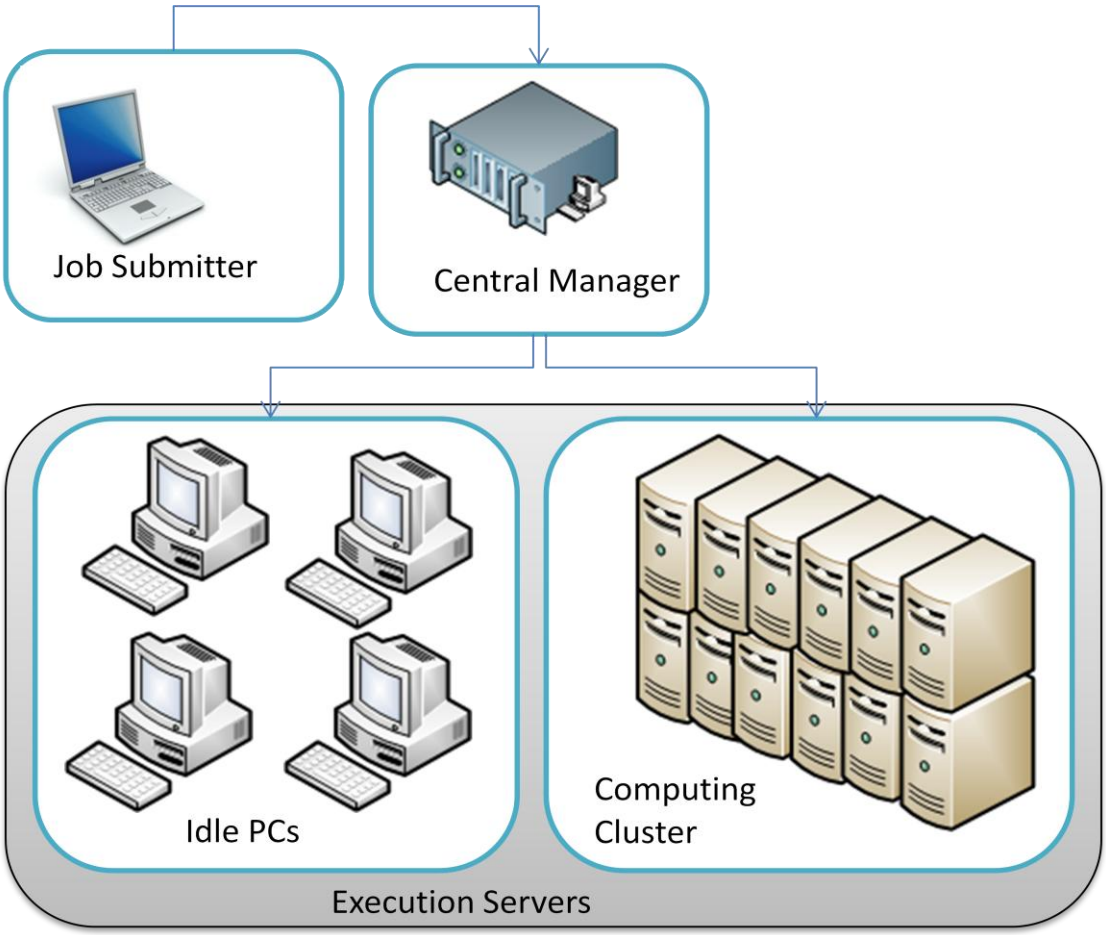

Figure 2.3: Grid diagram showing three major Grid Components

2.4.3.1 *Job Submitter*

The job submitter or user machine is a simple day-to-day computer used by a grid administrator or grid user to submit jobs on a grid (i.e. through a central manager/grid scheduler). The user needs to submit any job along with a submission script that specifies the distribution of tasks amongst execution servers and all other supporting files (e.g. libraries, etc...) that may be required during the program execution process.

2.4.3.2 *Central Manager*

The central manager is a control unit for the grid. This machine is really critical for security, because all grid servers and I/O operations are controlled by this machine. The central manager holds grid scheduler, resource discovery, authentication and authorization components that proves it to be a gateway to the grid infrastructure. It also

holds the information such as jobs execution status, the details of all execution servers, logs of operations, etc.

2.4.3.3 *Execution Servers or Worker Computers*

The execution servers are the worker machines that execute the actual task assigned by the grid central manager. These servers need to have required applications installed to execute the jobs. Figure 2.3 above shows two types of execution servers: a dedicated cluster of computers and some individual user computers. The individual user computers can only be used when they are idle, or a big portion of their CPU is idle. Nowadays, the new computers (such as Core2duo, Quad Core and Core i7 architectures) have more than one microprocessor: usually only one microprocessor is utilized and the rest are normally idle on average personal computers, these idle cycles can be used by a grid to perform grid based operations. The criterion for utilizing the grid resources is set by the grid administrator that resides as a policy on central manager.

A dedicated cluster of computers can be made part of grid executers when the grid of already available idle CPU cycles is not enough to accomplish assigned tasks.

2.4.4 The Grid Computing Resources

Grid computing allows communities to build virtual organizations all around the globe without any physical boundaries. The virtual organization can share their computation and storage resources with other nodes anywhere in the world. This also enables the researchers and organizations to share their resources with other peers and manage their resources from remote places. There are three main kinds of resources that are required to be handled in case of grid computing, namely computational resources, storage resources, and network resources [15]. The grid and peer to peer computing are emerging to solve the large scale scientific problem, which either requires more computational power, storage space or other network resources, or a combination of different kinds of resources. Below is a concise description of three common types of grid resources with their importance in a grid paradigm.

2.4.4.1 *Computational Resources*

The computational resources are the most important aspect considered in most of the grid systems, as the majority of the grid systems are built to achieve more computational power. The growing demand for computational power in new applications and research cannot be fulfilled with a single computer, especially when there are time restrictions to finish any computer intensive applications. Of course one can buy a blade server to accomplish this task but it is not easy for everyone to afford a blade server due to its hardware and dedicated software costs. Many scheduling algorithms and applications have been proposed to implement an effective grid resource broker (a grid component that dynamically distributes the resources amongst different jobs [20]) that can justify the organizational needs. The selection of a resource broker also depends on the kind of middleware used to implement the grid infrastructure [21] [22].

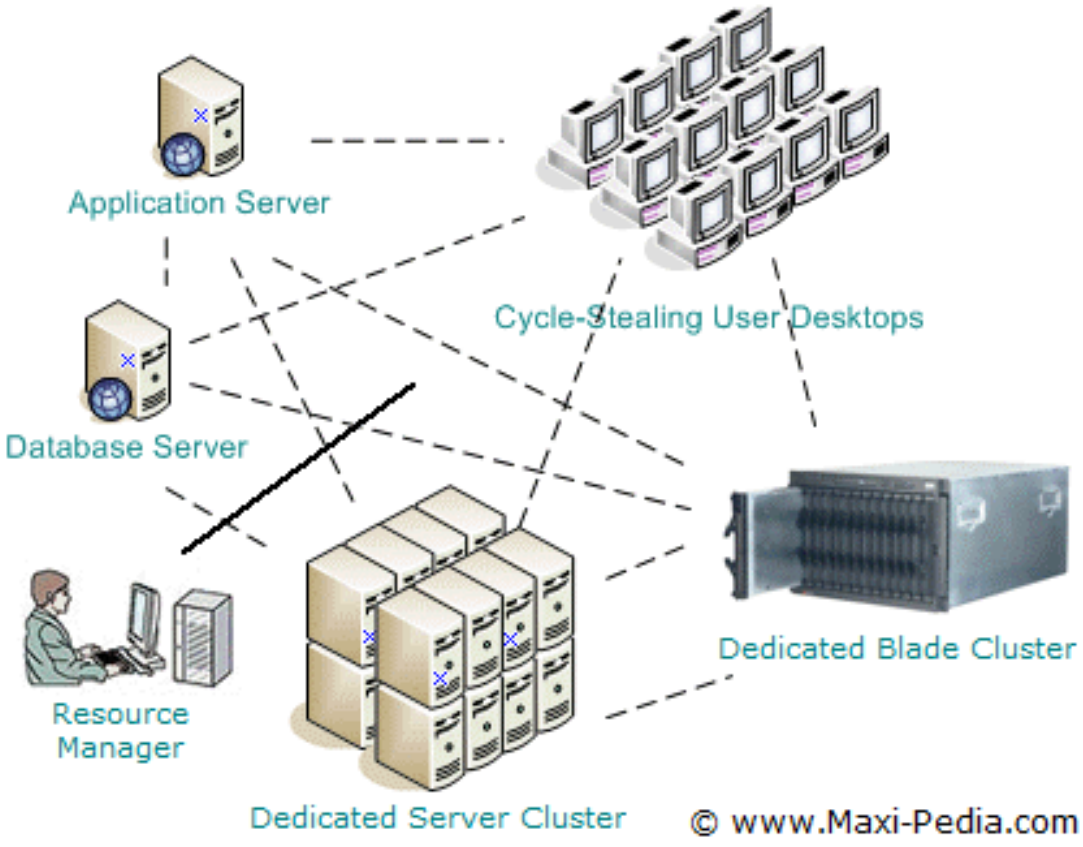

Figure 2.4: Demonstration of computational resources in a grid environment [23]

Figure 2.4 shows a computational grid infrastructure with different computational resources connected as a single network. This includes a dedicated desktop computing cluster, a dedicated blade cluster, stolen idle computational cycles from user's PCs, an application manager, database server and a resource manager. The resource manager holds responsibilities to distribute these computational and data resources between different grid users and applications on the basis of some scheduling algorithm (e.g. Priority scheduling or first come first served, etc.). Further Maxi-Pedia defines computational grid as

> "...the application of resources of multiple computers in a network to a single problem at the same time, while crossing political and theoretical boundaries." [23]

In summary the grid is about letting people share computing power, databases, knowledge, experience and other resources securely online across corporate, institutional, and geographic boundaries without sacrificing local autonomy.

2.4.4.2 *Storage Resources*

Grid computing is not only about sharing the computational cycles; it is also about sharing storage space and other hardware and software resources. An application designed to work on a computational grid often requires the allocation and co-allocation of many other resources, which are connected by network to achieve better hardware and software performance.

Many data intensive applications require huge amounts of data to be stored and processed. Projects such as GRIDPP (Grid for Particle Physics) [24] UK, Grid Data Farm [25] and SDSC (San Diego Supper Computing) [26] have built their own SRBs (Storage Resource Brokers) to deal with a different kind of database management system and repositories stored on various locations. The storage resource broker is an interface implementation to heterogeneous data sources, allowing easy access and management to various distributed data repositories.

2.4.4.3 *Network Resources*

Network resource management is concerned with allocating, de-allocating, monitoring and managing the network resources, such as an appropriate bandwidth allocation to files transfers, etc. Many applications cause high network bandwidth usage and while some places there is only a little or virtually zero bandwidth usage. These available network bandwidths can be shared with other programs that require more bandwidth, hence network bandwidth aware Job Scheduling is possible, and many grid computing infrastructures have been built to serve this purpose [27] [28]. These projects have designed special purpose bandwidth schedulers to meet the application requirements and avoid delays over network transfers.

2.4.5 The Grid Architecture

The grid architecture can better be defined as a layered structure, where every layer depends on the layer below. The user of grid computing accesses the grid using the topmost layer, which is normally an application layer while the bottom layer is a hardware and resource layer. Figure 2.5 below describes this generic concept, followed by Figure 2.5 that describes a detailed example of grid computing architecture.

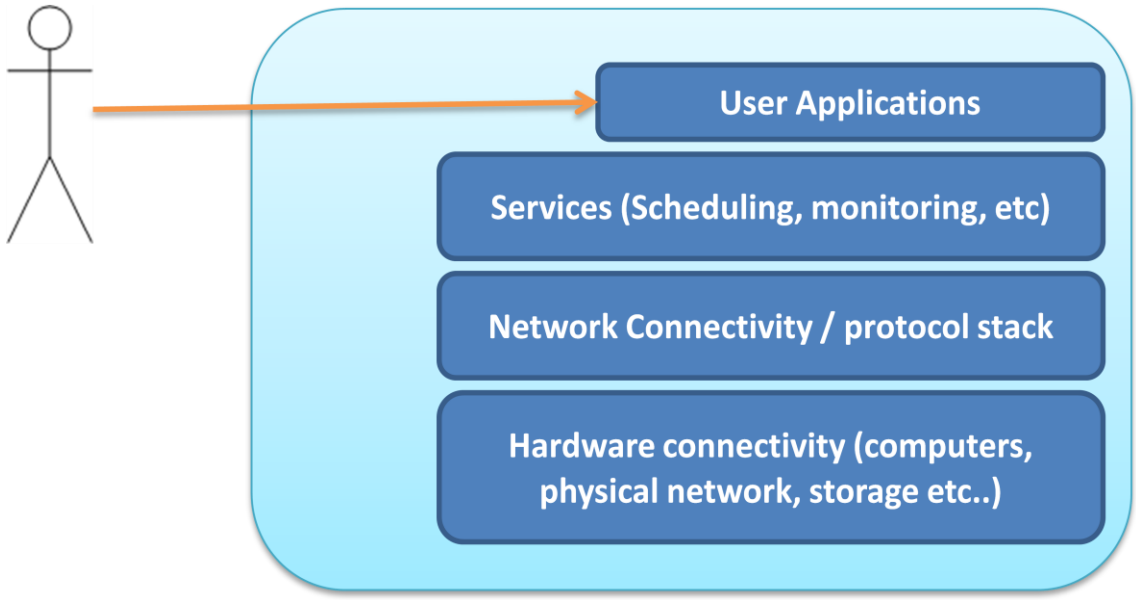

Figure 2.5: Grid Layered Architecture

The grid architecture shown in Figure 2.5 defines the layered architecture of grid infrastructure. All the layers provide input to the layer above, with the lowest layer being a hardware layer that includes computing and physical connectivity and where other computers or hardware devices are connected. As the grid is a distributed network based infrastructure, it relies heavily on a set of protocol stacks, such as TCP, HTTP and FTP on a very basic level, and many others on a higher level such as resource discovery, resource registry protocols, security layers, etc. The architecture of grid infrastructure also depends on its usage (i.e. type of applications it is built for) for example one kind of application may require more 3D graphics and animations to be rendered on the grid infrastructure. In this case we would need rendering software, frame distribution scripts, video encoders (in case of animations files), FTP protocols to upload and download the animation files, etc. On the other hand, if the applications are mathematical calculations based on a MATLAB or Physics domain, then various application packages and protocol stacks are required to execute these applications. However the grid developers try to cover more than one application and install all possible software and protocol stacks to make it more generic. Precisely every execution machine needs to have applications installed to execute the commands given by a central manager.

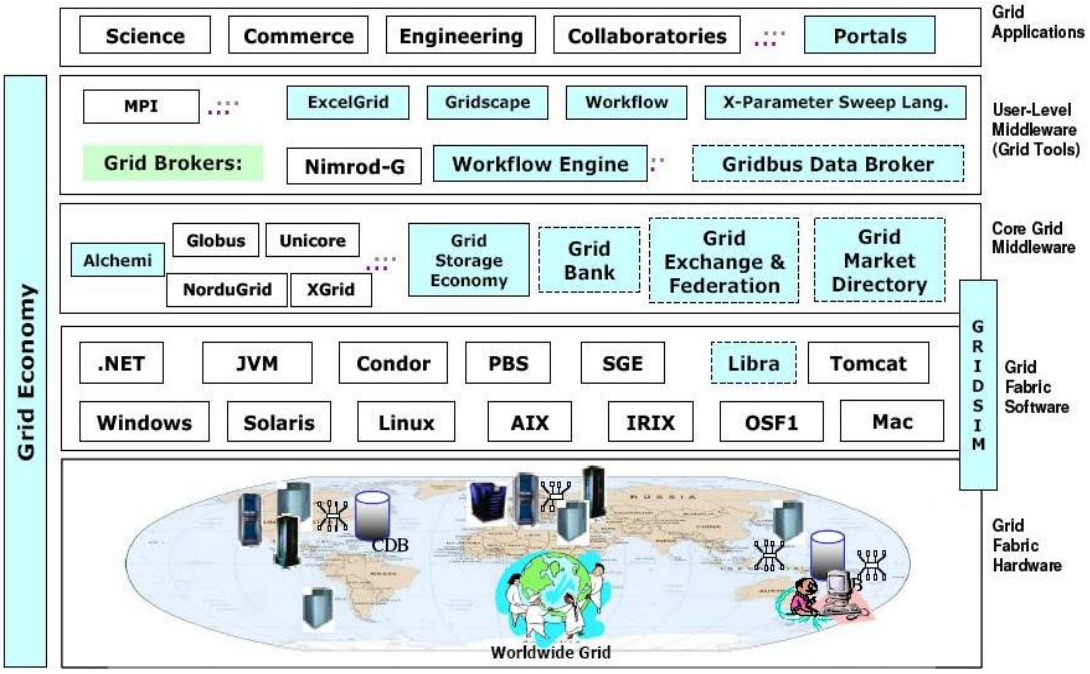

Figure 2.6: Grid layers infrastructure and architecture [29]

Figure 2.6 is a detailed layered infrastructure diagram, borrowed from the Cloudbus website [29], to demonstrate various software and hardware implementation possibilities in one grid infrastructure, i.e. Gridbus project. The diagram above shows many technologies, including some core technologies from Cloudbus (e.g. Alchemi [30] [31], Gridbus broker [32], GridSim [33], Gridbank [34], etc.) and others such as Condor [35], Nimrod-G [36], Globus [37], etc., all in one big grid infrastructure. The overall grid infrastructure is divided into different layers, separating hardware and software elements and also core software elements from user level applications.

Figure 2.6 shows five layers, where each layer depends on the layer below, and uses the interface provided to communicate with other layers. The first layer is the grid applications layer with examples of some well known disciplines where the grid is currently used. The layer-2 is the user level application layer where grid application software is listed; this application software can be used to access grid resources, and is an interface between user and actual core grid infrastructure. The layer-3 is the actual core grid middleware layer, which contains all grid functionalities and it manages all hardware and software resources using the interfaces provided by layer-4. The layer-4 is the operating system and system software layer where actual grid (layer-3) can reside; this layer also provides all interfaces to access the grid resources. All major grid middleware systems use operating system services (called system calls) to perform any operation and manage the grid resources. Finally the layer-5 is an actual hardware and resource layer that includes computer systems, network, printers, databases, and other grid infrastructures (as additional resources).

2.4.6 Grid Middleware Technologies

Grid middleware is a software package that acts as an interaction layer between software applications, actual computer hardware and network infrastructures. The middleware is usually accessed by users through a user application using some sort of API or an interface provided with middleware technologies; these applications are usually called grid portals. Generally the grid middleware technologies include job schedulers, security and protection mechanisms, match makers, job monitoring, resource monitoring, some set of APIs, and a command line or GUI interface to access

the grid environment. At present there are many freeware, open source and commercial grid middleware available on the Internet. Below is a precise description of some of the most important and frequently used grid middleware technologies, followed by a short conclusion to justify the selection of Condor high throughput computing environment for implementing the render grid proposed in Chapter 4.

2.4.6.1 *Condor*

Condor is high throughput computing system that makes use of idle CPU cycles to fulfil the requirements of intensive computing applications such as simulations, rendering, and other mathematical and scientific applications. The Condor tool monitors the machines that reside in a local network that has Condor software installed. When any of these machines becomes available, Condor adds them to the resource pool so that they can be used by currently running applications [38].

2.4.6.2 *gLite Middleware*

The gLite is a lightweight grid middleware developed by CERN (European Organization for Nuclear Research). The gLite middleware is mainly used by the European funded project called EGEE (Enabling Grid for E-scienceE) currently implemented in 12 academic institutions, Universities and industries across the Europe [39]. In addition to EGEE implementation of gLite, it is also implemented in many European and international research organizations. The gLite middleware mainly defines a grid infrastructure for batch-processing systems; the jobs are fed as batches with some priorities assigned to every job. The gLite middleware has been technically supported by EGEE and CERN and it allows its integration with many other grid middleware such as Globus, Condor, etc.

2.4.6.3 *Globus*

Globus toolkit [37] is one of the most popular grid technologies found in research, and is developed and supported by the Globus Aliance community [40]. It is an open source toolkit for building grid systems. The Globus toolkit contains security, resource management, data management and many other tools that enable the grid developer to

build a robust and portable grid infrastructure, and start writing applications for a grid using provided software packages. The Globus is a Linux based toolkit and, at present, most of the physics high power computing projects such as CERN high energy physics, GridPP (Grid for Particle Physics) [41], NGS (National Grid Services) UK [42], TeraGrid USA [43], and many other projects have used the Globus toolkit to build their grid infrastructure.

2.4.6.4 *Alchemi*

Alchemi [30] is a grid computing framework developed in the Microsoft .Net environment. It is a Microsoft Windows operating system based application that allows an easy implementation of computational grid infrastructures. The user can aggregate the computing power of computing resources connected through the Internet or local area network. As Alchemi is a Windows-based application it restricts it usage to Windows applications only, but due to the large number of Windows machines in many organizations, it enables Alchemi to create a computational grid by harnessing the idle CPU power, thus an easy to build and easy to use supercomputing solution. The Alchemi grid can be integrated with Globus-based grid infrastructures to support both Linux and Windows applications. The grid service broker (GSB) is responsible for deciding which application to run on what platform (i.e. it directs all Linux based applications to Globus grid and Windows based applications to both Alchemi grid).

Additionally, Alchemi provides a multi-thread programming API model, which allows users to build and execute multi-thread grid-based applications. Also due to its .NET API, the Alchemi applications can be developed in any programming language that is supported by the .NET platform (e.g. C, C++, C#, VB .Net, etc).

2.4.6.5 *UNICORE*

UNICORE (Uniform Interface to Computing Resources) [44] is an open source grid middleware system with additional guidelines to enhance the system for new functionalities and improvements. UNICORE software packages come in two kinds of components: server components for setting grid server, and client for setting worker machines. Additionally, UNICORE live CD for client components can also be

downloaded to run whole operating system (Linux) and grid clients from a CD ROM without the need to install anything on computer systems.

2.4.6.6 *Sun Grid Engine*

Sun Grid Engine (SGE), previously known as Codine (COmputing in DIstributed Networked Environments) [45], is an open source batch queuing system developed by Sun Microsystems. The SGE is a kind of a cluster computing environment responsible for scheduling the jobs, distributing the resources between the jobs, monitoring and managing the already running jobs in a Sun Grid Engine environment. Sun Microsystems also sells the commercial version of SGE, named as Sun N1 Grid engine, with technical support.

2.4.6.7 *OGSA-DAI*

OGSA-DAI (Open Grid Services Architecture Data Access and Integration) solution allows the federation of data repositories within a grid or cloud computing environment through web services. The objective of the OGSA-DAI project is data sharing on grid infrastructures locally in same network or globally over the Internet [46]. The OGSA-DAI has recently implemented a complete workflow management system for integration of OGSA-DAI with a computational grid to provide data access during workflow execution time, which is not supported by normal grid workflow systems [47].

2.4.6.8 *BOINC*

BOINC (Berkeley Open Infrastructure for Network Computing) is a program that lets you donate your idle CPU power and time to many charity projects such as SETI@home [48], Climateprediction.net [49], Rosetta@home [50], World Community Grid [51], and many others. Once the user has installed the BOINC component on his computer, he can then connect to any number of BOINC projects and donate his idle CPU power to these projects [52].

2.4.6.9 *Xgrid*

Xgrid is software developed by Apple's Advanced Computation Group for developing a grid computing environment for Mac-operating system users. This is similar to a normal grid on Linux and Windows; it allows group users connected in a network environment to share their computational power on a grid at very low cost. The Xgrid acts as a job scheduler for distributing jobs on available machines. Some of the successful implementations of Xgrid are Xgrid@Standford and BLAST projects [53].

2.4.6.10 *Why Condor*

The selection of the right middleware technology was critical as it needs to be compatible with both present requirements and future changes. Although for this part of work, Condor high throughput was selected, the output of this research is not limited to Condor only. The aim is to produce a cost effective rendering solution using grid computing to use the available CPU cycles for rendering purposes. Whatever middleware is chosen it needs to have the following capabilities:

- It needs to work on Windows as well as Linux machines
- The middleware should allow CPU Scavenging in local area networks
- It should be customizable, preferably open source
- The middleware should have good documentation, tutorials and reputation for regular updates and support.

For the above set criteria, Condor seemed to meet the rendering requirements along with the appropriate tutorials, documentation, regular updates and free user support through the Condor mailing list. Another inspiration to use Condor was a previous effort named as GDI|Queue [54], which was made to build a free rendering solution for 3ds Max, Maya and XSI. This is no longer updated since the 3ds Max 3.x version.

2.4.7 Grid Job Schedulers

A scheduler is one the most important components of grid computing, as all the distributed applications need to be assigned and distributed on many connected machines for execution. The grid scheduler is based on a predefined policy decides how

to distribute a job on grid infrastructure. A generic three-phase grid scheduling architecture defined in [55] is a better way to understand how a grid scheduler works. These three phases are further divided into smaller steps or subtasks.

1. Resource Discovery
    a. Authorization
    b. Application definition
    c. Minimum requirement filtering
2. System Selection
    a. Information gathering
    b. System selection
3. Job execution
    a. Reservation
    b. Job submission
    c. Monitoring
    d. Job completion
    e. Cleanup

When a job is submitted in a grid computing environment for execution, it requires many resources to complete its execution: the scheduler verifies whether the job requested for a resource has access to that particular resource or not. Every application has minimal limitations that need to be specified by the user, limitations such as operating system, hardware and software requirements, etc., are set up during the job definition.

The next step is to filter out the non-required resources so only minimal required resources are left to execute the job. There may be many similar resources that can meet the same minimal requirements. During the second phase, all the resources are examined and only one resource for one requirement is selected based on an algorithm.

During the third phase, required resources are reserved for faster execution and no delay. A job is submitted to the grid system, which is then distributed on different machines for execution. During the execution the job is regularly monitored either by the end user or it is automatically monitored by the grid system. Once the job is finished

the end user is informed and the cleanup process is executed; this releases all the resources currently occupied by the running processes.

2.4.8 Grid Resource Allocator

Grid resource allocator is a grid component that works closely with the grid scheduler; it reads the required resources for a newly created job, selects the appropriate resources, and allocates them to a job. Once the job is finished, the grid resource allocator de-allocates already assigned resources.

2.4.9 Grid Monitoring

Grid job monitoring is an interface for the user to check the status of job execution and resources in a grid environment. Typically, the grid monitoring tools are command line, supplied with grid toolkits such as Condor_q and Condor_status commands with Condor to check the status of jobs and machines. There are many third-party grid monitoring tools available on the web for easy to access and understand the grid execution such as GridView [56], GRAM (Globus Resource Allocation and Management) [57], etc.

2.4.10 Grid Portals

The portals are the interfaces to the grid computing environments, and their main objective is to provide an easily accessible interface for end users of the grid, hiding the detailed difficulties of the actual grid from novice end-users. Although the grid middleware provides some sort of interface to access grid services, typically these interfaces provided by grid middleware are difficult to understand, especially for new users (i.e. many are UNIX or DOS based interfaces). This may be due to the grid middleware developers concentrating more on grid functionalities than the grid interface. Using a third party user interface such as P-GRADE (Parallel Grid Run-time and Application Development Environment) [58], AHM [59], etc., provides an easy access to core level gird infrastructures. Due to recent developments in grid computing, the grid is becoming as easy to operate as a computer user operates his web based email box, or any other web portals. The grid portals are used by many popular grid

infrastructures to provide additional and easy access to its users, for example, P-Grade is currently used by EGEE Grid, South-Eastern European Grid, NGS UK Grid, Knowledge Grid Malaysia, CLGrid Chille, etc. [60]. Building a grid portal is not as difficult as it could be, if one had to write everything from scratch in a C / C++ programming language; it is becoming more and more easy as new software development kits specifically written for developing grid portals such as Grid Portal Development Kit (GPDK) [61], etc., have been introduced.

2.5 Some Popular Grid Projects

Grid computing is emerging due to many new projects that have been implemented in the past few years. Many national and multinational projects have been granted funding by many large funding bodies. Organizations such as the European Union, CERN and many others have fully supported grid computing in the past few years. Below follows a brief explanation of the major grid technology projects that are relevant to the innovative render grid proposed in Chapter 4. However for various reasons these grid technologies are unsuitable for implementation of the render grid proposed. An example of why they might be unsuitable could be simply due to their large scale nature, in which case it is practically impossible due to resource access to the IT infrastructure at the University of Sussex to contemplate installation of major grid technologies. This is in fact the subject of a working group at Sussex – in other words the IT infrastructure at Sussex has a bearing on what can be practically implemented that is also innovative and useful. The innovative solution in Chapter 4 relies on being able to be deployed in a museum with little IT infrastructure and policy. Nevertheless, it is interesting to review these grid projects to see what they have been used for and how this may feed into the innovative solution render grid presented in Chapter 4.

2.5.1 EGEE

EGEE (Enabling Grid for E-scienceE) is a European grid computing project [62]. The purpose is to build a 24hour online grid computing infrastructure for scientists, researchers, and academia across Europe. The EGEE grid infrastructure is one of the largest infrastructures with 40,000 CPUs and several Petabytes of storage across 250

resource centres that currently support 13,000 researchers worldwide. The middleware technology installed on EGEE infrastructure is gLite [39], developed by CERN laboratories. Besides gLite middleware, EGEE supports and recommends many additional applications in combination with gLite for easy access, monitoring and workflow systems, such as GridWay Meta scheduler [63], P-Grade Portal [60], and GANGA (Gaudi /Athena and Grid Alliance) [64] front end interfaces for job submission.

2.5.2 TeraGrid

TeraGrid [43] is a US-based grid infrastructure coordinated through the Grid Infrastructure Group (GIG) at the University of Chicago. It is mainly used for scientific discoveries in many disciplines: Molecular Biosciences, Physics, Chemistry, Astronomical Sciences, etc. The TeraGrid infrastructure has greater maximum computing and storage capabilities than any other grid with more than a Peta flops computing and 30 Petabytes storage capabilities. The TeraGrid project has a similar infrastructure to the UK NGS grid (i.e. Globus middleware). Also there has been a successful collaborative project named "Teragyroid", implemented between TeraGrid USA, UK NGS and RealityGrid in 2003, the purpose was to explore lattice-Boltzmann simulations involving lattices of over one billion sites [65].

2.5.3 NGS

NGS (National Grid Service) [42] is a UK-based grid computing project funded by the EPSRC (Engineering and Physical Science Research Council) and JISC (Joint Information Systems Committee). The NGS is led by the STFC (Science and Technology Facilities Council) e-Science Department, and other major partners include the University of Edinburgh, University of Manchester, University of Leeds, and University of Oxford. The aim is to support UK researchers HPC and collaboration across the country and with many other countries, the NGS grid is connected with EGEE Europe and TeraGrid USA. The middleware technology used by NGS is Globus and SRB (Storage Resource Broker) for data management.

2.5.4 GridPP

GridPP (Grid for Particle Physics) [41] is a collaboration of particle physicists with computer scientists from UK-based Universities, the Rutherford Appleton Laboratory and CERN. The GridPP is built to store huge amounts of data from various projects being carried out in different parts of the world; these data will be analyzed and archived by the GridPP project.

2.5.5 WLCG

WLCG is the Worldwide LHC (Large Hadron Collider) Computing Grid designed by CERN for High Energy Physics community to handle the huge amount of data generated by large Hadron collider [66]. The LHC expected experimental data is around 27TB with an additional 10TB summary data. The grid is connected to more than 170 institutions and it is expected that the data produced by LHC in experiments will add up to 15 Petabytes of data each year [67].

2.5.6 BRIDGES

BRIDGES (Biometrics Research Informatics Delivered by Grid Enabled Services) is a UK e-Science project coordinated by the University of Glasgow. The project is about developing and exploring database integration over six geographically distributed research sites within the framework of the large Welcome Trust biomedical research project, Cardiovascular Function Genomics. BRIDGES aims to test both the OGSA-DAI (Open Grid System Architecture, Data Access and Integration) and IBM Information Integrator to test the performance of integrators [68].

2.5.7 Climate*prediction*.net

Climate*prediction*.net is the largest experiment to try and produce climate forecasts in the 21st century. This project uses the idle power of others' machines when these machines are switched on and are not in use [49]. The climate*prediction*.net uses the BONC Berkley Open Infrastructure for Network Computing [52].

2.5.8 SETI@home

SETI (Search for Extraterrestrial Intelligence) [48] is a scientific area whose goal is to detect intelligence outside the world in space. The SETI@home uses the radio telescope to listen for narrow bandwidth radio signals from space. Originally the SETI project had a dedicated supercomputer installed with a telescope to deal with the bulk of data analysis. In 1995 it was proposed to use SETI@home to build a virtual supercomputer through CPU scavenging from donated CPU cycles using the BOINC tool.

2.5.9 Monash SPONGE Facility

Monash SPONGE facility uses Condor high throughput computing to use the idle CPU cycles for research projects. This Monash SPONGE is part of the Monash Campus grid [69]; it uses the existing computing machines at the University of Monash. During working hours these machines are mostly busy but as soon as they become free, their power is used as a SPONGE facility. The SPONGE facility provides a single interface that can be accessed by 'sloth1.its.monash.edu.au' using SSH (Secure Shell) or any equalling tool.

2.6 Motion Capture Systems

The use of motion capture technologies to provide innovative solutions for digital heritage systems, in particular in the field of embodied interaction with a digital heritage environment is presented in Chapter 5. The research work carried out here is focussed around the eMove – Personal Motion Sensing System project whose main goal is to develop a low cost consumer level upper body motion capture suit for the gaming industry. Mapping this technology into an embodied interaction scenario within digital heritage environments is the focus of Chapter 5. In this respect, technology developed in eMove by the author of this thesis is adapted to heritage presentation and interaction services as described in Chapter 3, the presentation of DISPLAYS. The following sections discuss various human motion capture technologies, their benefits, similarities, differences and reasons for selecting the eMove motion capture system over other available options.

Most graphics and animation applications require more attention in their visual impact, so digital heritage presentation and interaction systems require this visual impact to maintain user interest. Motion capture technology, when linked in real-time to an avatar in a digital heritage world, is visual and interactively stunning. For this reason, the ability to captivate the heritage visitor to the virtual heritage world, motion capture technology is proposed for interaction services in the DISPLAYS framework. Whether we talk about a decent video game or TV advertisement, a cartoon show or a 3D movie, animated puppet or sports training, a military application or a scientific simulation, all require more attention in their look and feel, special effects and behaviour than real life characters. A simple frame by frame animation can take months to create a powerful and realistic 3D real life animation in a virtual character; on the other hand the same animation can be generated in few minutes once the character and environment are already set up. Thanks to motion capture technologies which can input real life action directly into a 3D avatar on the fly, it allows the user to create an exact copy of real life animation into a virtual environment for virtual characters.

Figure 2.7: Image from 'Avatar' a 3D movie released in 2010.

The movie 'Avatar' used 3D graphics, animation and real-time motion capture data heavily. This movie had the highest income in the history of the film industry [70] [71]. *The image show: Zoe Saldana, an actress wearing motion capture suit (on left hand side), acting as 3D digital avatar (on right hand side of the picture), image source:* [72]

Motion capture, which is also referred as “motion tracking” or “mocap” is all about capturing the positions of a human or any object in 3D space, recording it and translating these movements into a virtual 3D character.

> “Motion capture is a process of obtaining and recording three-dimensional representation of live action performance or even by capturing an object’s position and / or orientation in physical space” [73] .

> “Motion capture is the process of recording a live motion event and translating it into usable mathematical terms by tracking a number of key points in space over the time and combining them to obtain a single three dimensional representation of the performance.” [74]

The motion capture hardware (which could be a motion capture suit, digital camera or magnetic field) produces the raw data; essentially the position of an object in 3D space is recorded to a medium, which is further refined by some software filters so that it can be used by any end user applications. The cleaning process involves reducing jitters/jerky movements/noise from captured data; this is either done in real time or in the post-cleaning process. Many motion capture companies produce their own solutions to remove/reduce these jitters from motion capture data, also software such as Autodesk MotionBuilder [75], which works as an end user application for 3D animation, provides a built-in jitter filter named Butterworth filter [76] to clean the noise or spikes from mocap data.

The motion capture systems work in two modes:

1. On-line or Real-time mode, called Real-time motion capture systems
2. Offline mode, called offline motion capture systems

In on-line systems, the motions of a real object are captured in real-time, the movements and/or positions of the performer are analyzed and instant feedback is produced to the end user. This feedback could be in the form of animation playing in accordance with the motion capture data, or it could be controlling some physical objects such as a robot,

etc. This all is done in real-time without any significant delay, and the process also includes a data cleaning process using some sought for jitter cleaning filters.

The second form is an off-line motion capturing system, where the motion data is captured and stored in some kind of standard such as BVH (Bio Vision Hierarchy) format or flat files. The stored data can then be used later to perform animation with post effects. These kinds of systems are particularly useful where the user applications require post processing in animation data, such as cartoons, movies or some game effects, etc. The offline motion capture systems usually do not provide any real-time or instant feedback to the user of mocap devices.

Both of these have different applications; the first one (i.e. real-time motion capture) is mostly used in real-time simulations and gaming applications, and the second (i.e. Offline motion capture) in cartoons, 3D movies, etc.

2.6.1 Types of Motion Capture

There are various methods found in the literature and software commercial market to capture the position of a human or any object in 3D space. These methods vary in terms of accuracy, performance and cost; the user needs to select the appropriate type of motion capture technology that meets the project requirements in terms of cost, performance and deliverables. The motion capture technologies generally break into three main types, namely Optical, Magnetic, and Mechanical motion capture systems. Below is a discussion on these motion capture types, with details on how these systems work and kind of technologies they use. Brian Windsor et al. give a decent overview from an artist's perspective on motion capture systems in his book "MoCap for Artists" [77].

2.6.1.1 *Optical Systems*

The optical motion capture systems rely on markers placed strategically on the human body which are tracked by a multi-camera system, typically 8 to 16 or more cameras are used to track whole body motion. The systems can only detect the position of a marker in 3D space although the rotation can be calculated with some post

processing/calculations. There are two kinds of markers used in optical motion capture systems and the method of operation varies according to the kind of markers used in the motion capture process. Hence there are two sub-types of optical motion capture systems: 1) Reflective marker motion capture systems, also called passive motion capture systems; and 2) Pulsed-LED, also called active markers motion capture systems.

The main advantage of the optical system is that the motion capture data is clean/noise free: the main disadvantages include a dedicated studio space is required to set up cameras, the light interference may cause problems, and the system can only operate in limited studio space (i.e. studio or room space).

2.6.1.2 *Reflective Markers Motion Capture Systems*

The Reflective Marker Motion Capture Systems [78] require a number of reflective dots placed on a performer's body; these dots are tracked by multiple cameras in the surrounding environment. All cameras used in the set up have Infra-red LEDs (Light Emitting Diodes) mounted around the camera lens, along with IR pass filters placed over the camera lens. As the dots are reflective, the position of these dots can easily be tracked by adjusting the camera threshold to avoid skin or clothing so that only the reflective dots are sampled. The popular examples of reflective motion capture systems are Vicon [78], [79] and Motion Analysis [80].

Figure 2.8: Passive Motion Capture System:

A person wearing markers, walking on stage, on top of image, many cameras with red LEDs can be seen, also a projector screen showing a digital format of stage with 3D character. Image source: [81]

2.6.1.3 *Pulsed-LED Markers Motion Capture Systems*

The optical systems based on pulsed-LEDs require LED markers to be placed on a human/object's body segments. The cameras can detect the position of these LED markers by measuring the Infra-red light emitted by the LEDs rather than light reflected from markers. The popular vendors for pulsed-LED maker motion capture systems are PhaseSpace [82] and Optotrak Certus [83] motion capture systems.

Figure 2.9: An actor wearing active markers on different body segments

Behind him is a projector screen showing his 3D digital avatar. The little digital cameras, connected by cables can also be seen on top right side of the image. Image source [84]

2.6.1.4 *Magnetic Motion Capture Systems*

The magnetic motion capture systems involve magnetic receivers strapped on the human body which is surrounded in a magnetic field through a centrally located magnetic transmitter. Each receiver is connected to a separate interface and all are synchronized; these receivers can measure their relationship with the transmitter, this results in the data stream from receivers consisting of both position and orientation into 3D space.

Figure 2.10: An example of magnetic motion capture

A person wearing magnetic motion capture suite, the image also shows a magnetic field behind the person wearing the magnetic suit. Image source [85]

The main advantage of magnetic motion capture system is that the data produced by the system is accurate and do not require any further calculations. The disadvantages include the fact that the output data is usually noisy and requires cleaning, the system is prone to magnetic interference (even the cement built floors may cause interference to the system), also the data sampling speed is too low for many fast applications such as sports activities, etc.

2.6.1.5 *Mechanical Motion Capture Systems*

Mechanical motion capture systems are usually designed as an exoskeleton suit that can be worn by a human. In mechanical mocap, the angular data of joints (e.g. elbow, shoulder, hand, etc.) are measured from the sensors placed on the human body instead of positional data. The exoskeleton or prosthetic structure contain angle measuring devices such as potentiometers are linked via rods, and all the rods are connected to one

another building a whole human body skeleton. The skeleton design issue limits the use of mechanical systems to particular size or limited adjustable sizes only, i.e. the exoskeleton suit designed for a skinny person needs adjustment to fit a fat person.

The major vendor for mechanical exoskeleton systems is Animazoo UK LTD [86] as they hold major patents in this area. Their popular exoskeleton systems are Gypsy 6 full body and upper body suits [87] and IGS-190 full body suits [88].

Figure 2.11: Full body gypsy 6 exoskeleton suit from Animazoo, image source [89]

Mechanical systems also have advantages and disadvantages, for example they cannot provide the position of body in world space without supplementary technology, but they do give rotational angles directly for joints. This is advantageous in applications requiring angular data only (note optical systems have to calculate angles from position data), but positional data can be computed from angular data and knowledge of the skeleton hierarchy (forward kinematics). The cost of mechanical systems is considerably lower and they are less prone to environmental interference than marker based and magnetic systems, they also offer virtually zero latency.

The innovative solutions proposed in Chapter 5 adopt the eMove upper body motion capture suit used input device for digital puppetry – in fact the full 4D TRAVEL scenario [90] uses the Animazoo IGS 190 [91] [92] for the digital heritage main

character and the eMove suit for visitor embodied interaction. The main reason for using eMove suit is its accuracy, cost and availability for research as part of the eMove project.

2.6.2 Motion Capture Industries

Just a few years ago motion capture was quite expensive and was only used in a few highly specialized industries or group of researchers. In addition, it also requires high level skills to operate motion capture hardware and software. The whole mocap system was too sophisticated to be used by a common person. The recent developments in motion capture industries have given a huge scope to motion capture technologies: now most of the cartoons, games and many 3D movies rely heavily on motion capture for quicker animation and real life effects.

At present there are many big names in motion capture industries and commercial markets for selling motion capture systems. Below are some of the popular hardware and software manufacturers for motion capture systems.

- Vicon systems
- Motion Analysis
- PhaseSpace
- Optotrak Certus
- Animazoo
- Xsens MVN Motion Capture
- Microsoft Kinect Project

The first four (Vicon, Motion Analysis, PhaseSpace, Optotark Certus) are marker based systems, Animazoo are mechanical suit vendors, Xsens is also inertial motion capture technology, the same as mechanical motion capture except the sensors are strapped on body segments rather than connected on rods, the Xsens also provides a full body suit with sensors, i.e. ‘gyroscope’ embedded, this is same as Animazoo IGS -190 suit. See Figure 2.12 showing both suits side by side.

Figure 2.12: On right a female wearing Animazoo IGS-190 suit [86], on the left side a female wearing similar full body motion capture suit from XSens [93]

The Microsoft Kinect is a new name in motion capture systems (expected to be released by the end of 2010) specifically designed for the gaming and entertainment industries. It is kind of optical motion capture, gesture and voice recognition system where no markers are required to capture the motion data. It also works with only one digital camera instead of the 8 to 16 digital cameras in typical motion capture systems [94].

2.7 Lip Syncing and Facial Animations

Lip syncing is an important part of a digital puppetry solution, as it provides a real life feel to character animation. Below is a list of the most important 3D character animation and lip syncing solutions found in the literature, starting from MotionBuilder that has been used inside our current solution. Some of the solutions only provide voice phonemes that can be fed to any animation software such as games engines to produce lip animation, also the facial expressions can be controlled by key board or joystick keys.

At present MotionBuilder has been used to build digital puppetry solutions to demonstrate the concept, and the reason to choose MotionBuilder is that it is the only solution that supports real-time lip-syncing and facial expressions along with motion capture data. The other solution presented below is either non-real-time, or they are

even more expensive than the current solution. More research is required to develop a cost effective solution for lip-syncing and facial expression, see future work in Section 6.4.

2.7.1 MotionBuilder

MotionBuilder [75] is an impressive high quality real-time character and facial animation solution that supports lip-synchronization based on Voice Reality phoneme-extraction technology. Due to its real-time features, it is more suitable for interactive 3D applications and its high quality graphics and animation makes it more suitable for 3D movies, virtual cinematography, and other performance animations.

MotionBuilder provides a software development kit (SDK) for developing plug-in and small applications inside MotionBuilder. This allows the developers to create additional plug-ins and functionalities inside MotionBuilder. Its voice reality technology allows users to easily produce real-time facial animation and lip animation synchronized with audio inside a 3D environment.

2.7.2 Blender 3D

Blender is an open source 3D graphics and animation tool, supported by the Blender foundation [95]. Blender features include armature (skeleton) deformation with forward/inverse kinematics with pole target support, built-in lip-sync solution based on Blender fundamentals such as vertex and face loops, sequencer and Shape Key system. At present all of the seen examples of character animation based around Blender lip-syncing uses audio from wav files rather than direct input from a microphone. More research needs to be done in the Blender area to find a more cost effective digital puppetry solution based around Blender 3D, and voice to phoneme extraction software such as Magpie.

2.7.3 Magpie

Magpie is a desktop application a product of Third wish software and animation [96]. Magpie provides features such as automatic speech recognition, real-time preview and support for lip-syncing and facial animation. It provides a very accurate 2D, 3D and

Stop Motion facial animation, which can be incorporated into existing animation packages like 3ds Max, Maya, Blender, etc. However, it allows real-time facial animation into the Magpie environment and exports it into other applications as an animation file.

2.7.4 iClone

iClone 3 is similar to Magpie and does not support real-time lip syncing. The recent demo videos of iClone 4 show some possibilities of using real-time facial expression and lip-syncing inside iClone [97]. More research needs to be done for motion capture data and real-time audio based lip-sync inside iClone to find a cost effective solution compared to our original solution based on MotionBuilder.

2.7.5 Crazy Talk

Crazy Talk is similar to Magpie, and provides lip-syncing for recorded audio files only [98].

2.7.6 Papagayo

Papagayo is open source software for lining up phonemes (i.e. mouth shapes) with the recorded sound of a user [99]. We have not found any example that tells us about any possibility of implementing Papagayo for real-time lip syncing.

2.7.7 Source Game Engine

There are quite a few game engines, such as the Source game engine [100], that support motion capture, lip syncing, facial expressions and game assets all together in one place.

The source game engine requires additional software packages:

1. Voice recording software,
2. Face Poser (comes with source SDK)

The workflow for creating facial animation inside source game engine is:

- The human voice is recorded using appropriate voice recording software.

- The recorded audio is then imported to the face poser for phonetics definition,
- Then a human model is imported inside the face poser
- The face poser then analyzes and generates lip movements
- Resultant lip animation with character and recorded voice can then be imported inside the source game engine for its presentation in game environments.

We have tried to implement a real-time solution but could not figure out with Source game engine. The Source game engine workflow clearly shows that it requires additional software called Face Poser that does not allow any real-time lip-syncing for the source game engine. Also up to now we have only seen examples that demonstrate predefined audio recording, phonetics definition and then facial animation.

2.7.8 FaceAPI

FaceAPI is an SDK developed by Seeing Machine to detect head movement and facial expressions using a normal digital camera [101]. FaceAPI provides a programmable interface to access the data from FaceAPI to stream and use inside any other applications. Although FaceAPI is a commercial SDK, Seeing Machine also provides a non-commercial version for educational use that can be used to test requirements. However the educational version is only available with head tracker data and does not include lip syncing and other facial expression features. The educational version comes with demo [.exe] application that can be used to demonstrate the accuracy of lip-syncing and facial expressions on demo screens but one cannot extract the actual data from educational version.

For the research for innovative solutions presented in Chapter 5, FaceAPI was used to implement the data streamer and head tracker inside the Unity3D game engine. The head tracker works up to 90+/- degree as claimed by the Seeing Machine website. However, the actual lip-syncing data from SDK could not be accessed due to demo version limitations, nevertheless from observations of the ‘demo.exe’ application it was

found to have two problems that are critical to successful lip synching for a digital heritage character.

- The demo version provided with FaceAPI SDK showed that the lip-sync solution has certain limitations that will not allow a 3D avatar to stretch the mouth for a big smile or big 'O' sound, etc.

- Another secondary issue is the overall cost to build an application. For any application including a digital heritage scenario – licenses were simply too expensive for practical bespoke system design. This issue relates to MotionBuilder, because MotionBuilder does provide an effective solution for lip syncing, however a MotionBuilder license is about $4,000 per use in any system. Any other solution (i.e. the development with an API such as the FaceAPI) has to beat this MotionBuilder cost to make it worth the effort to implement the solution.

2.7.9 Annosoft SDKs

Annosoft provides software development kits (SDKs) to allow a developer to develop their own application using phonemes produced by either wav files or audio direct from a microphone in real-time. It offers two kinds of voice-to-phoneme conversion software development kits [102] :

1. Annosoft Lipsync SDK
2. Annosoft Microphone SDK

2.7.9.1 *Annosoft lip-sync SDK*

This SDK version is an offline/non real-time solution that requires users to record their voice into wav files. The files are then imported inside the tool that adopts Annosoft lip sync SDK e.g. a C++ program, etc, the SDK retrieves the phonemes, which the user can process (i.e. apply to a cartoon character). This solution does not support real-time voice capture from microphone to SDK, hence is of no use to a real-time digital heritage character animation, such as that outlined in Section 1.4.2 'Scenario: 4D TRAVEL', or

more specifically Section 1.5.2 ‘Heritage interaction scenario using motion capture based avatars’.

2.7.9.2 *Annosoft Microphone API*

The Annosoft microphone API (or real-time SDK) is more relevant to the requirements for the ‘4D TRAVEL’, ‘heritage interaction scenario using motion capture based avatars’, and other scenarios needing real-time lip-syncing. The real-time SDK will allow lip-syncing with real-time motion capture based animated characters (or avatars) to be implemented as a part any other application, e.g. game engine, etc. The developer can utilize phonemes generated in real-time to drive visemes created inside 3D tools to trigger different blended facial motions including morphing between visemes. This SDK is a more suitable solution, but it costs around $5000 per year for a license that may not be suitable for digital heritage objectives. Although if the number of units to be developed is more than two or three, then Annosoft may be the correct choice for building an alternative to the existing MotionBuilder solution.

2.8 Game Engines

As a part of the research presented in this thesis, care needs to be taken on the cost of a digital puppetry solution for digital heritage systems that museums can afford, thus ideally open source components should be used; freeware or at least favourable licensing conditions for software components. The research has looked at and implemented several solutions with the Panda3D engine (open source), Unity 3D (limited version free to use, better versions have favourable licensing models), Unreal (favourable licensing). The selection of any of these (and there are many others that could be considered) bring any digital heritage system that needs a graphics visualisation with appropriate interaction down to a more affordable cost. To achieve a digital heritage walk through objective, these game engines have been tested with the eMove motion capture system.

These three popular games engines are described in more detail below. For each a new motion capture suit driver has had to be developed because this is a first for the integration of motion capture to a gaming engine in this context, i.e. the ability to walk

through a digital heritage environment using real-time mocap coded in a gaming engine. So, the inclusion of a more affordable voice-to-phoneme SDKs, similar to Annosoft or FaceAPI (discussed above) to allow lip-syncing with real-time voice integrated inside these game engines, will provide a more rounded innovative digital puppet solution. See Chapter 5 for digital heritage museums and sites for kiosks and virtual presentations. Additionally, other animation techniques such as morphs or stop motion, will allow various facial animations. There are hundreds of games engines available for download on the Internet [103], choosing the right game engine with the right requirements is important. For this research some set of criteria should be set, such as:

The selected game engine should have:

- Scripting capabilities with scripting interface
- Skeletal Animation/bone animation support
- Preferably open source else freeware or low cost
- Availability of tutorials, documentation, and community support
- Morphing or facial animation support

For the above set criteria, the three games engines mentioned above were chosen, starting from open source Panda3D [104], then freeware Unity3D [105], and then commercial Unreal for different experiments for real-time motion capture drivers. At present the open source Blender is also being tested for real-time motion capture drivers, lip-syncing, and facial animations. Note, these engines are not necessarily the best options but they were selected because they fulfil the above set criteria.

2.8.1.1 *Panda 3D*

Panda3D is an open source gaming engine developed by Disney VR for creating 3D graphics and attractions for their major theme park applications. The Panda3D supports bone animation and python scripting that allows real-time motion capture data to be connected with an animated 3D avatar. The other useful features of Panda3D include collision detection, actor animations, scene graph exploration tools and 3D audio.

2.8.1.2 *Unity3D*

Unity3D is a powerful game engine, at present free for personal use and free for commercial use, except in case of commercial distribution that needs to be licensed. The current version Unity gaming engine offers post-processing effects, advanced physics engine, vast terrain rendering, shedders and audio features. The Unity bone animation and C#, Boo and Java script based scripting facilities allows connecting real-time motion capture data with 3D avatars inside unity engine. An integrated editor allows the user to make dynamic changes while playing inside the game environment; this is really handy for test purposes. The Unity provides suitable documentation and video tutorials for novice users to start building games faster than other traditional gaming engines.

2.8.1.3 *Unreal*

Unreal is a game engine developed by Epic Games [106]. It is one of the most popular games engines at present as it can be observed from a large list of games developed using the Unreal game engine [107]. The Unreal 3 game engine is free for non-commercial use with all functionalities of a commercial engine; this allows developers to create their games and test them before actually buying it. The powerful engine with powerful rendering, animation, audio, physics, etc. facilities and regular updates, prove Unreal to be one of the next generation games engines. The motion capture system has been tested inside the Unreal engine; it also works fast and smooth with excellent graphics.

2.9 Summary

As explained in Chapter 1, existing digital heritage systems are lacking in many useful technologies for their content rendering and interaction services compared to existing ICT technologies. Hence there are many useful technologies that can benefit digital heritage in its content creation, presentation and interaction services. This chapter was focussed around a description of these technologies, their functionalities, types, alternative solutions, and differences between proposed and alternative technologies.

The first part of this chapter discussed the technologies involved in a content rendering service, i.e. service orientation and grid computing, their technical descriptions, functionalities and architecture. The service-orientated architecture has been proposed as implementation architecture for the whole DISPLAYS framework; this also includes render grid implementation (part of this thesis). Moreover, the Condor high throughput computing environment that is used for implementation of the render grid, has been described along with its alternatives, followed by the justification of using Condor over other solutions have been described.

The second part this chapter was about motion capture systems that are proposed for heritage content interaction service for the DISPLAYS framework. This part highlights the various solutions, alternatives, and justification of using eMove motion capture technologies over other available solutions. This area is also required to work with 3D animation environments where the interaction can be presented and for building an audio and video chat application. Various 3D animation environments, e.g. games engines are introduced along with audio and video technologies for chat application, which is also built as a part of this research.

CHAPTER III

3 DISPLAYS Framework

This thesis proposes a conceptual framework (in effect a requirements gathering framework) called DISPLAYS (DIgital library Services for PLAYing with Shared heritage resources) or developing a set of services for **creating**, **interpreting**, **using** and **exploring** cultural heritage resources by museums and user communities. The main innovation behind DISPLAYS is the development of a service-oriented architecture where five main services are proposed, namely, Digital Content: Creation (DCC), Archival (DCA), Exposition (DCE), Presentation (DCP) and Interaction (DCI) services, all connected by a workflow management system focused on the creation, interpretation, use and exploration of digital heritage resources and objects, which can be implemented from scratch or composed of existing components (provided they are service oriented). Figure 3.1 illustrates the proposed DISPLAYS framework.

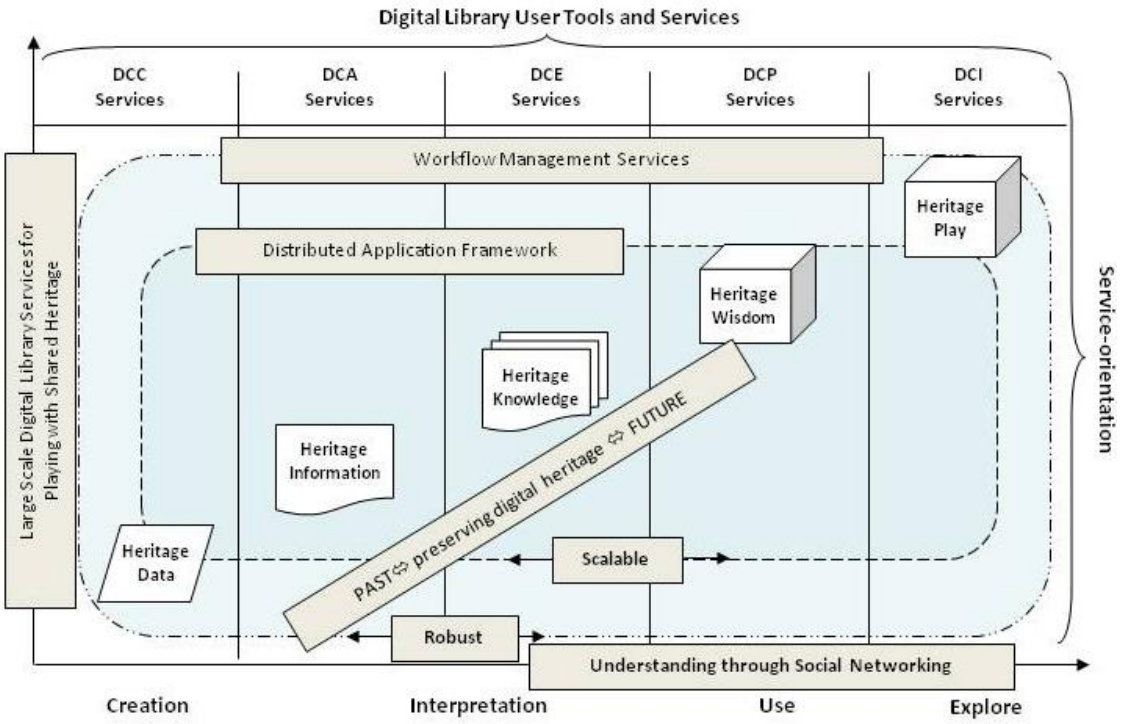

Figure 3.1: DISPLAYS services framework

Figure 3.1 illustrates that these services (or tools depending on actual implementation detail) are each associated with certain workflows, for example:

- *DCC Services* (or tools) are associated with the creation of *heritage data*, e.g. the creation of digital heritage resources. Examples could be a simple digital representation of a cultural heritage artefact along with its metadata and textual description from a museum catalogue. To create such a digital object simple existing tools could be deployed, e.g. MSWORD to digitize the text according to an agreed metadata standard [108], a digital camera and Photoshop to prepare the image, and so on. More sophisticated tools include 3ds Max, Maya, etc., to prepare 3D content, and the render grid proposed in this thesis to prepare digital heritage scenes.
- *DCA Services* are associated with managing heritage data, organizing that data into collections (archiving) that imply greater *heritage information* such as 'this object belongs with these other objects for whatever curatorial reason'. Thus, these services rely on *interpretation* from museum experts, and there may be many interpretations defined by the museum experts. Obvious examples here

include a database system configured to provide at least the standard Create, Read, Update and Delete (CRUD) functions, but also other administrative functions, a notable example being the ARCO database system.

- *DCE Services* are associated with the generation of *heritage knowledge*, which relies again on museum curators and other such experts organizing the managed knowledge into expositions (or exhibitions) that provides a particular *use* of a collection of artefacts (digital objects). For example, a collection of Sierra Leonean masks or a collection of Romano British objects from an archaeological site. For this function a more sophisticated content management system than just a simple database to archive objects is needed. Again ARCO has such a CMS (Content Management System), but other open source museum based CMS are also becoming available [109], [110].
- *Presentation services* are associated with the way a collection of digital objects is displayed in a heritage context to the user or community. Examples are as a virtual online museum, or a museum kiosk, using simple Flash-based technology through to augmented reality relying on standard server technology through to specialized servers that can handle VR and AR services. Again, the ARCO ACMA presentation manager provides an example [4]
- *Interaction services* are associated with different interaction methods that allow communities to play with digital heritage resources and objects. Thus, communities can explore such objects in innovative ways that are far more interesting than the simple observation of museum catalogues online. An innovative example of this is the implementation of online real-time motion capture technology to create a digital puppet that enacts a virtual historical character to explore heritage stories proposed in this thesis.

Thus, the DISPLAYS concept provides all necessary functionalities – depending on the implementation methodology – for a digital heritage resource to create, use, interpret and explore a collection, or multiple collections, of digital heritage artefacts or objects so that these objects can be of greater benefit to the community by exploiting its services or tools. However, a DISPLAYS framework can be implemented in many ways, for example, a particular DISPLAYS framework could simply be composed of a

partial ARCO system that implements DCA, DCE, and DCP components (or services) with an additional interaction (DCI) service (see Section 1.5.2 'Interaction through motion capture'), or it could be a completely new implementation built from scratch, such as the 'validation architecture' (i.e. the reanimating cultural heritage system currently being implemented [5]). The DISPLAYS service orientation approach also allows systems to be composed from other existing modules made available from other digital library systems, e.g. BRICKS, using web services. Whichever implementation method is used, the idea is that both community users and museum professionals alike can access digital heritage objects and content in an interactive and engaging way – a primary goal.

The DISPLAYS concept also proposes to use existing social networking technologies such as Flickr, YouTube and Facebook in an innovative mashup to share community generated content with other communities and museums based on the context of the museum collection exhibited or presented in the DISPLAYS system [111]. The DISPLAYS concept (or requirements specification) is validated to some extent by the implementation of a real-life application scenario called the 'reanimating cultural heritage digital resource'. This is funded by the 'AHRC Beyond Text' programme (the project is called 'Reanimating cultural heritage: digital repatriation, knowledge networks and civil society strengthening in post conflict Sierra Leone' – or more simply 'reanimating cultural heritage', with the acronym RCH for short) [5]. The current version of RCH project (under construction on the time of this thesis) can be found at ["http://reach.rn.informatics.scitech.susx.ac.uk/sierra-templates-v9.9-dev/"].

An interesting feature of DISPLAYS is the concept of classifying the architecture into five parts, which builds on ARCO [112], which itself was classified into four parts. DISPLAYS breaks up the concept of 'exhibitions' (ARCO's fourth part) into 'presentation' and 'interaction' components to allow a more flexible consideration of interaction technology (i.e. separates out interaction as a component in itself), and how it integrates with the 'presentation' of digital heritage content (objects and scenarios of use for these objects).

Chapter 5 presents a novel use of 3D digital heritage objects and real-time animation using motion capture technology (so called 'Interaction through motion capture' mentioned in Section 1.5.2) for unique, high quality and engaging interaction within heritage scenarios in more detail. This thesis also presents an innovative component for the DCC services (see Chapter 4), which proposes a render grid utilizing cost effective grid computing solutions to facilitate the inexpensive generation of heritage virtual scenes [113] [114].

Before looking at other digital heritage libraries or systems, it is useful to summarise the main goal of DISPLAYS. The DISPLAYS services system seeks to bring both communities and professionals together to facilitate knowledge networking – where, for example, a museum curator can discover facts about an artefact from a village elder, and the community can access their heritage, currently located in museums, through a digital repatriation approach. Thus, a main objective of DISPLAYS is to provide a digital library system around a set of services that allows communities of practice such as museums, archaeological societies, commercial archaeological units, local history groups, metal detector societies, individuals and organizations on a large scale, to archive, share and use their archaeological and historical data, knowledge, understanding and wisdom in a safe, accountable and self-preserving way. As mentioned above, this DISPLAYS concept and in particular, this goal is put into practice with the validation architecture, i.e. the RCH digital resource [111].

How does DISPLAYS compare with other similar systems? There are of course, other digital library projects that exhibit some of the features specified for DISPLAYS, but none provide the total set of features. Some of the most notable results relevant to the DISPLAYS framework are found in the DELOS, BRICKS, DILIGENT and ARCO projects (others relevant projects are discussed in Chapters 4 and 5 where they specifically relate to the innovations presented there), and of these ARCO is now commercialised with the University of Sussex and Poznan University of Economics holding the IPR for this system. Table 3.1 provides a baseline set of features for each of these systems compared against the proposed DISPLAYS framework. This table does not explore each system in great detail; rather it gives a succinct overview to illustrate the main goals of each system and the features needed to achieve that goal.

Table 3.1: Comparison of several key digital library systems features against DISPLAYS

Services	**BRICKS**	**DILIGENT**	**DELOS**	**ARCO**	**DISPLAYS**
Heritage Goal	G1	Yes G2	G3	G4	Yes (G5)
Digital Heritage Library Implementation	Yes	Yes	No	Yes (Complete System)	Yes (RCH Validation Systems, other components
Content Creation Services	Yes (ECLIPS-RCP)	No	No	Yes	Yes
Archival Services	Yes	Yes	Yes (demonstration only)	Yes	Yes
Exposition Services	Yes	No	No	Yes	Yes
Presentation Services	Yes	Yes	Yes (demonstration only)	Yes	Yes
3D Interaction Services	Yes (ECLIPS-RCP)	No	No	Yes	Yes
Technologies	**BRICKS**	**DILIGENT**	**DELOS**	**ARCO**	**DISPLAYS**
Service Orientation	Yes	Yes	Yes	CMS Only	Yes
Social Networking	Yes (BRICKS only)	Yes (DILIGENT only)	No (DELOS only)	No	Yes (Cross platform)
Mashups	No	No	No	No	Yes
Virtual Reality	Yes	No	No	Yes	Yes
Augmented Realty	No	No	No	Yes	Yes
Virtual Exhibitions	No	No	No	Yes	Yes
Interaction techniques	No	No	No	VR and AR	VR and Motion Capture
Rendering Services	No	No	No	No	Yes

Each of the digital heritage systems or frameworks[1] has a main goal as indicated in Table 3.1. It is discussed further here in terms of a brief description for each system, its goals and outcomes as far as they affect or are useful to the DISPLAYS framework:

BRICKS Description: BRICKS (Building Resources for Integrated Cultural Knowledge Services), is an EU 6th framework funded project for digital heritage libraries. BRICKS is an integrated project, structured in three main areas of work:

> *Infrastructure:* A networked system of services, based on open standards, that integrates heterogeneous collections of digital multimedia documents.
>
> *Application services:* Four main application scenarios have been defined: access to culture, management of culture, creation of culture, and edition of digital texts. This makes a significant value for the involved user communities, and will demonstrate the market potential.
>
> *Sustainability:* One of the project's principle goals is self-sustaining in the future. The BRICKS Factory will define, develop and maintain a user- and service-oriented space to share knowledge and resources in the cultural heritage domain. The other technologies include semantic web, web services, DRM systems, watermarking, etc.

An online community has been developed, "BRICKSCommunity.org", in order to build consensus and a critical mass, to share knowledge and services for digital content, and to exploit the results achieved. The BRICKS community includes members that are content providers, art professionals, art researchers, students, and other interested users in general [3].

---

[1] The term framework in this context is meant to mean concept, specification, or architecture that is not necessarily implemented as a whole or in part, examples being DELOS and DISPLAYS.

BRICKS Goals (G1): is to build "a new generation of 'digital libraries', to be read as a comprehensive term covering digital museums, digital archives and other kinds of digital memory systems. The mission of the 'BRICKS factory' is to define, develop and maintain a user- and service-oriented space to share knowledge and resources in the cultural heritage domain" [115]

BRICKS Outcomes: if we assume BRICKS was completed and made available as an open source system, its infrastructure and particularly its application services would be extremely useful for developing a DISPLAYS implementation. However, it remains to be seen if BRICKS application services are actually available as open source at the date of writing this thesis. Certainly, at the time when the research was conducted for this thesis little detail and actual understandable code was available. Emphasis was therefore placed on developing the DISPLAYS framework, which could utilise the BRICKS component if it became available.

DILIGENT Description: is a Digital Library Infrastructure based on Grid Enabled Technologies (DILIGENT) and is also an EU funded project (6th Framework) for creating "an advanced test-bed that allow virtual e-Science communities to share knowledge and collaborate in a secure, coordinated, dynamic and cost-effective way" [2]. DILIGENT is aimed at a cost effective solution based on existing EU Grid infrastructure, and focused on heritage community collaboration. DILIGENT is a proposed test-bed, for testing wide area digital heritage library applications and their components on an European gLite grid infrastructure. The test-bed will be built by integrating Grid and Digital Library (DL) technologies for next generation e-Science knowledge infrastructure with many different research and industrial applications.

DILIGENT Goals (G2): the main objective of the DILIGENT project is provide a cost effective test-bed environment to enable e-science members to collaborate and share knowledge and other resources such as archives, database, software tools, etc., with other members on a wide area network using Europe's existing grid infrastructure.

DILGENT Outcomes: the outcome of the DILIGENT project will be a knowledge infrastructure (i.e. a collaborative test-bed environment) by the integration of digital library services with the existing grid infrastructure.

DELOS Description: DELOS is a network of excellence on digital libraries; it is an EU-funded project whose aim is to build a bridge for integrated and co-ordinated research efforts in regard to digital libraries in European Union countries. It also provides a theoretical and practical framework for the evaluation of digital libraries and their components. The DELOS vision is to enable any citizen to "access human knowledge anytime and anywhere, in a friendly, multi-modal, efficient and effective way, by overcoming barriers of distance, language, and culture and by using multiple Internet-connected devices" [1]. The main goal for DELOS is research whose results are carried out by the participating community and shared between the participating researchers as well as with the general public in European countries.

DELOS Goals (G3): the objective of DELOS is to define a digital library system management reference model describing expected characteristics, functionalities and architecture of a digital library management system.

DELOS Outcomes: according to DELOS there are no standards for building digital library systems, therefore the outcome of this project will be to produce a digital library management system framework that can be used as a guideline to produce new digital library systems.

ARCO Description: the ARCO system is used for building virtual museum exhibitions. This system was developed in the ARCO EU 5th Framework project, led by the Computer Graphics Centre at the University of Sussex. The system allows museum professionals (and others) to create, manage, and build virtual exhibitions of digital museum artefacts or objects, and present them online in a virtual museum or exhibition, including the integration of virtual and augmented reality [4] [116] as a method of interaction. A virtual museum implemented with ARCO can be composed of a set of simple web pages with embedded 3D contents (3D museum artefacts) that can be dynamically updated from the ARCO database. The virtual museum could also be a virtual environment embedded on a web page, allowing users to navigate the environment while selecting and examining 3D objects.

The ARCO system (illustrated in Figure 3.2) has a number of good functionalities including a content management system that includes: a cultural object manager (a

cataloguing system), a presentation template manager (templates define the layout of the virtual museum on the web pages) and a presentation manager that defines the organisation of the virtual museum's exhibition spaces.

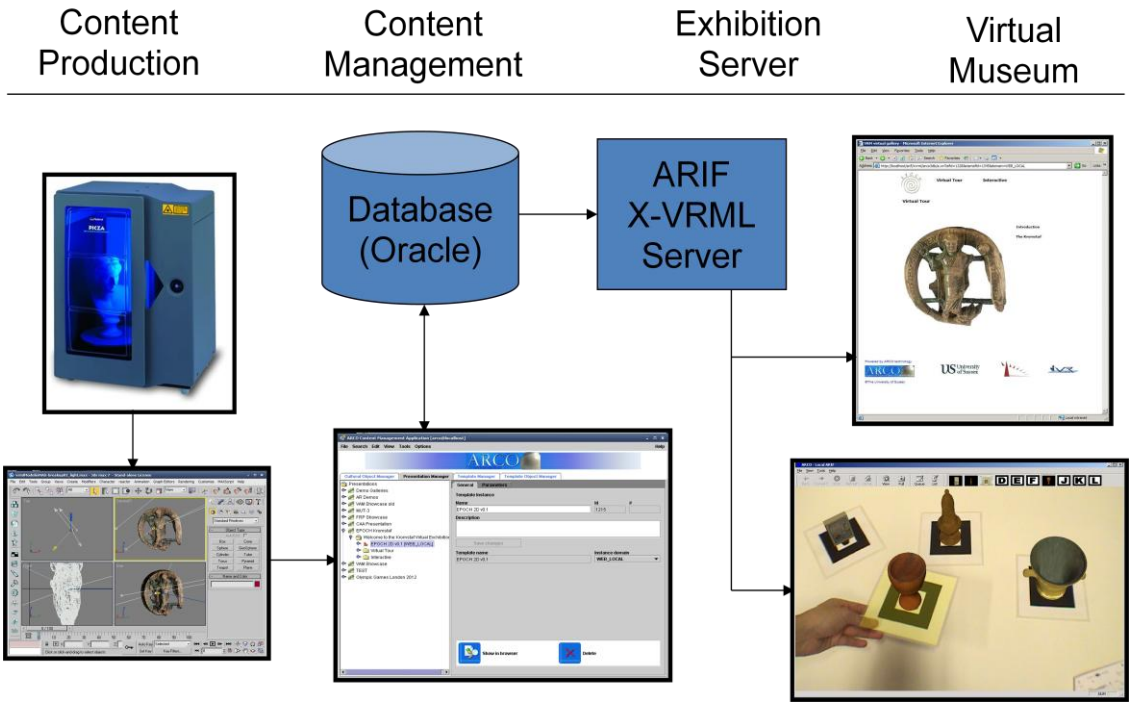

Figure 3.2: ARCO system showing various functionalities of ARCO [4]

Here we see the original four functionalities: content production (creation), content management (archival), exhibition (exposition), and virtual museum (presentation). The ARCO system is, however, intended for use by museum professional, giving them the capability to produce the end result, i.e. a virtual museum. DISPLAYS looks at extending this concept to include, amongst other things, better access and integration of community knowledge with more advanced interaction components. One particular technology that migrates from ARCO into DISPLAYS is the metadata element set based on Dublin Core and the mda Spectrum to facilitate collections management, and an XML Data Exchange format that allows import and export of data from other museum systems.

ARCO Goals (G4): a primary goal of ARCO was to create a system that could be used by museums to build their own virtual online exhibitions with their digital heritage objects.

ARCO Outcomes: although the ARCO project is now commercialised, it is still available for research purposes; this allows us to use ARCO components as part of a DISPLAYS implementation. For example the ARCO Content Management Application (ACMA) with its Object and Presentation Managers can be used via web services to implement the DISPLAYS Archival Services (DCA), Exposition Services (DCE) and Presentation Services (DCP). Figure 3.2 illustrates the ARCO functionalities, individual components, software and hardware interfaces and overall workflow.

DISPLAYS Description:

DISPLAYS Goals (G5): the main goal of DISPLAYS is to provide a conceptual frame that proposes a set of digital library services that will allow various communities of common interest (for example, archaeologists, metal detectors, museum curators and other organizations, and societies) to create, archive, share, use and explore cultural heritage knowledge in a safe, accountable and user-friendly manner.

DISPLAYS: Outcomes: the overall outcome for DISPLAYS is to provide a framework or concept upon which individual or groups of digital heritage components bases on either a web services or tools based approach can be linked through defined workflows. Example implementations include a first prototype system built around the ARCO system in the reanimating cultural heritage project, and the current 'reanimating cultural heritage digital repository'.

3.1 DISPLAYS

A more comprehensive discussion is now given for DISPLAYS. This description could serve as the requirements gathering exercise from which to specify a set of requirements for a particular DISPLAYS implementation. As already mentioned, DISPLAYS can theoretically be implemented in many different ways, thus this specification is necessarily incomplete.

The DISPLAYS concept is an evolving architecture and has its roots in the ARCO (Augmented Representation of Cultural Objects) system. ARCO does, however, have its limitations; for example, it uses a non-standard approach (i.e. XVRML templates) to

develop presentation services. XVRML is basically an XML wrapper around VRML to allow connectivity to the database and dynamic update of 3D contents [113]. Thus, you need to be an XVRML expert to create exposition and presentation components or services, of which there are few about. On the other hand, DISPLAYS advocates more standardised technologies for exposition and presentation, i.e. XML, XSLT, XQuery, etc., and in particular advocates the use of open source CMS systems to manage digital heritage content. On the other hand, ARCO does have a very good virtual and augmented reality system for building good interaction scenarios.

DISPLAYS is both a general concept and an architectural framework for utilizing different technologies and methodologies to benefit the heritage community at the maximum possible level. In other words DISPLAYS provides methods and techniques utilizing up-to-date and powerful hardware and software technologies (such as Motion capture, grid computing, mashups, social networking and workflow management), which can be compared with any other powerful ICT applications for ease of access, storage, collaboration and system management. The concept of DISPLAYS has evolved during the past three and half years of research in the area of digital heritage and other relevant technologies. The Computer Graphics Centre has worked on different components of this concept to build a framework and architecture with various implementation examples [117] [7] [118] [114] [119] [6] [111] [113] [120]. The proposed DISPLAYS framework currently includes several sub-component prototypes, some of which have been implemented in a validation version called the ‘reanimating cultural heritage repository’.

As Figure 3.1 illustrates, the DISPLAYS concept is based around the philosophy of refining heritage data created via content creation tools and services into useful heritage information through the use of efficient archival services or tools. This heritage information is further refined into heritage knowledge using exposition services that organises, for example, collections into online exhibitions. This new heritage knowledge base is thus organised into heritage displays, e.g. virtual museums or museum kiosk interactive systems, to benefit different communities and museum visitors. These heritage communities can then play or interact with their heritage. Thus, DISPLAYS provides a virtual but sensible and meaningful digital heritage environment

for museums to expose and present their knowledge to the rest of world using the proposed DISPLAYS heritage services. The immense power and variety of interaction services that are possible will allow the heritage community to interact, play and feel digital heritage objects in versatile and meaningful ways that will give a real experience of involvement inside heritage resources and a clearer image of the past within the present to learn interactively about their history.

DISPLAYS proposes the utilisation of cutting edge technologies such as:

- service orientation for easy access and portability of web services on every platform [6];
- a workflow management system for overall management of the system for easy access and tightening the set of relevant services into one management control [7] [117];
- the power of grid computing for rendering high resolution 3D graphics and animations [113] [114];
- a database management system such as the Oracle based ARCO system and other social networking repositories for archiving and preserving digital heritage resources;
- back-up and recovery solutions such as LOCKSS (Lots of Copies Keep Stuff Safe) for data and digitized heritage resource backup; and
- the use of web 2.0 technologies such as mashups and social networking to present and share heritage contents with other relevant communities and communities of the same interest [111].

A particular innovation of DISPLAYS includes the use of hardware technologies such as motion capture suits for interaction services. In this respect, this thesis proposes a unique solution for the creation of digital heritage characters that has been tested with the eMove exo-skeleton suit [121] to navigate through different gaming engines inside 3D virtual environments. It can be shown that the gaming environment can easily represent a digital heritage scenario (or heritage visual narrative rather than the standard game) where the motion capture technology (i.e. eMove exo-skeleton suit) can be used to navigate through the heritage environment embedded in the gaming engine. This

would allow the user (visitor wearing an eMove suit) to interact with digital heritage objects, thus experiencing these objects in a totally new and innovative way.

The critical innovation here, which is demonstrated in this thesis, is the real-time connection of motion capture technology (the eMove suit) to a gaming engine. In this respect several games engines have been tried for which real-time motion capture drivers have been written as part of the innovative research work presented in this thesis. Games engines explored are Panda 3D [104], Unity 3D [105] and Unreal [122] connected with the eMove exo-skeleton suit, and massively multiplayer online gaming (MMOG) server infrastructures such as RakNet to demonstrate a wide area heritage interaction concept. Additional components that can be integrated into a DISPLAY architecture include UDP based audio, video and a text chat interface (called eMove-chat) [119], which will allow heritage communities (e.g. a museum visitor) to participate in an embodied immersive interaction with a heritage story. This interface (motion capture integrated with audio, video and text) also allows community members to talk with other community members while interacting with a digital heritage world. Chapter 5 will discuss this innovative interaction technology to prove the overall concept.

As already discussed, elements of the DISPLAYS concept are implemented through the reanimating cultural heritage system [111], where several important DISPLAYS concepts are embodied. Figure 3.1 shown above illustrates the overall DISPLAYS framework, tools, services and system management. To summarise Figure 3.1 again before a more detailed description, the elements shown in middle of the diagram namely Heritage Data, Heritage Information, Heritage Knowledge, Heritage Display, and Heritage Play are the output of DISPLAYS services, where services are described as:

- The ‘DCC services’ => produces the heritage ‘data’, e.g. a collection of images or 3D object models or 3D heritage environments. A more concrete example is the ‘reanimating cultural heritage’ collection, i.e. the validation architecture.
- The ‘DCA services’ => that stores data along with metadata and produces ‘information’ archived inside heritage repositories. Examples are the ARCO database system, or the ‘reanimating cultural heritage’ XML flat file repository.

- The ‘DCE services’ => brings the ‘knowledge’ for the end users through different kind of expo or exhibitions created by a museum officer or an authorized administrator of the system. An example is the use of the exhibition manger technology in the ARCO system, or the organisation structure of the ‘reanimating cultural heritage’ prior to presentation, e.g. the XML file structure.
- The ‘DCP services’ => allows the museum curator to ‘display’ its contents over the web using various available options. Examples are the use of XVRML in ARCO, or the XSLT code and web server architecture in ‘reanimating cultural heritage’.
- The ‘DCI services’ => provides 3D software and hardware tools to the end user to ‘play’ with heritage objects and heritage environments. An example is the 3D scrap book or the social network functionality in ‘reanimating cultural heritage’.

This chapter is a brief discussion of the DISPLAYS framework as shown above, and the following sections discuss the concept around this framework.

3.2 DISPLAYS Services

There are five main digital library services proposed in the DISPLAYS framework. These are content creation services, archival services, exposition services, presentation services and interaction services.

Figure 3.3 illustrates the concept of DISPLAYS services (DCC, DCA, DCE, DCP and DCI services), where various technologies are listed in different services (vertically); the figure also shows possible workflows and services connectivity through various colourful lines (horizontally). This section will describe the concept of each proposed service in detail.

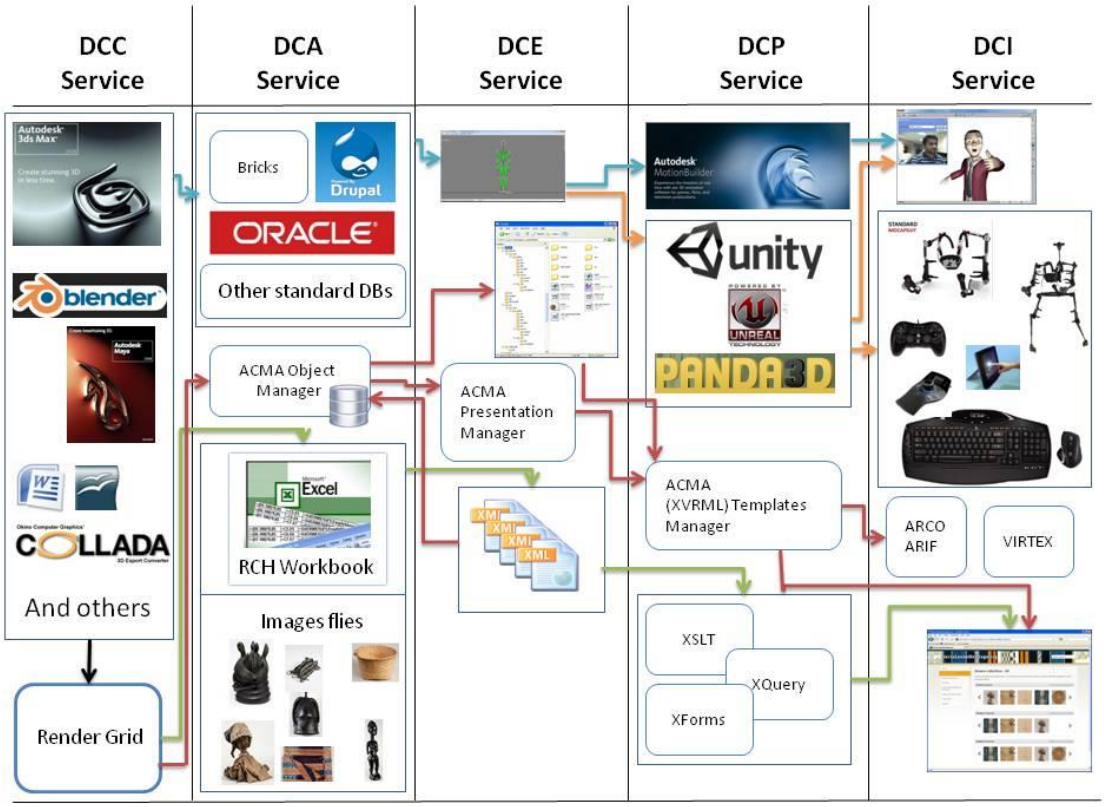

Figure 3.3: DISPLAYS services diagram with tools and services connected to demonstrate the possible workflows

3.2.1 DCC Services

DCC services are about facilitating content creators of digital heritage objects, e.g. allowing 3D modellers to work more efficiently, effectively and collaboratively with other 3D modellers [118]. This service can also be implemented as a part of 3D authoring tools such as 3ds Max, Maya or Blender (open source), etc. This service also proposes the exchange of digital contents such as 3D objects, heritage models, environments and documents over the network in a standardised and interoperable way. As an example service, an innovative render grid has been implemented [114] that allows 3D modellers to use spare machine capacity as a render farm when they are idle, see Chapter 5. Additionally, the innovative eMove-chat application, briefly mentioned above, can be used for conversation during the collaborative developments.

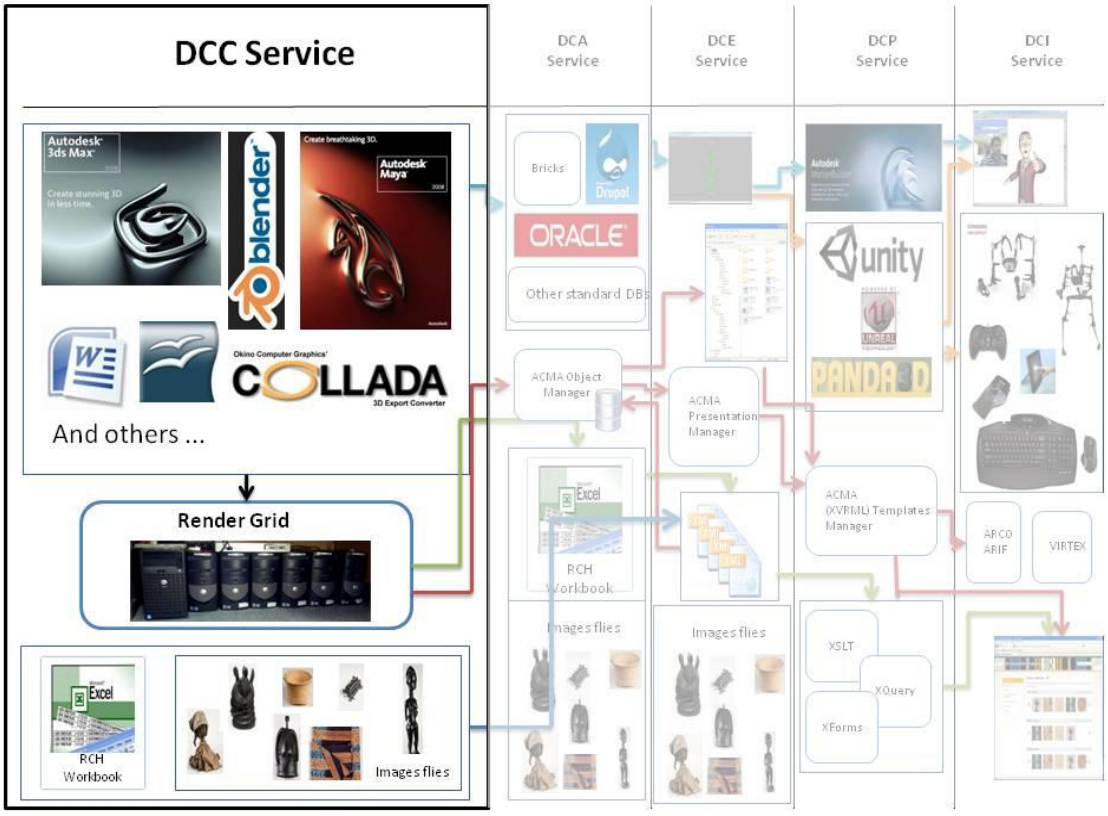

Figure 3.4: DISPLAYS Services diagram with expended DCC service

Establishing a collection of associated content creation services or tools allows many domain experts to work together with 3D modellers during the design and modelling phase of 3D digital heritage resources and objects designed for heritage visualization and interaction scenarios. For example, a museum education officer can remain in the development cycle of modelling a heritage object by monitoring and suggesting changes during the development cycle through his own attached terminal somewhere in a remote location. For another example, imagine the scenario where 3D modellers, art historians, museum curators and IT experts need to collaborate in the building of a museum interactive based on a 3D reconstruction of the Church of Santa Chiara. Victoria and Albert Museum art historians and museum curators have access to knowledge embedded in hundreds, maybe thousands of documents and images that inform them, and hence the 3D modeller, on how best to build the 3D reconstruction. Both 3D modeller and museum experts can work together to build the 3D church model by collaborating with a set of DISPLAYS content creation services or tools. Collaborative efforts in this respect include sharing of knowledge, e.g. documents such as images,

text, photographs, drawings, sketches and explanations, even videos of existing objects in the museum and on location.

Figure 3.5 illustrates the proposed 3D collaborative environment architecture for a digital content creation service exploiting a render grid (implemented in Chapter 5).

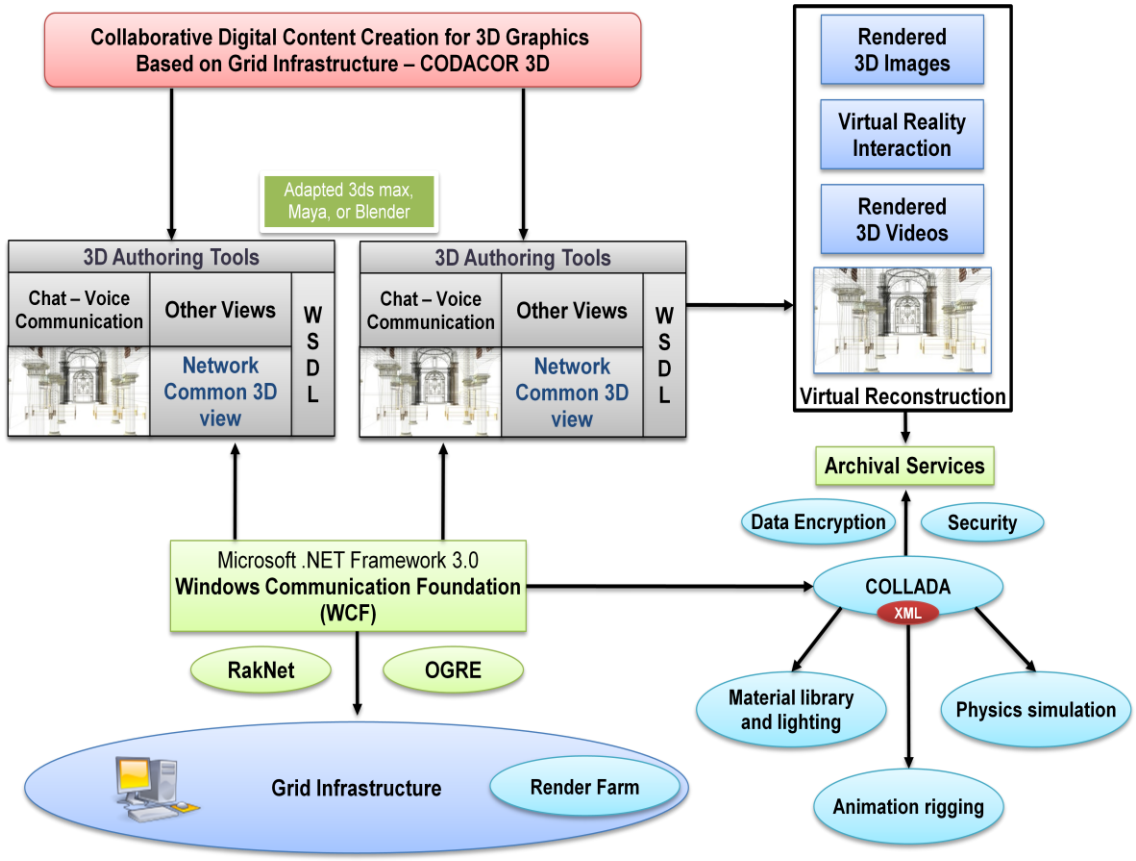

Figure 3.5: Collaborative 3D DCC exploiting a render grid

Using Figure 3.5 as a basis for a set of high level requirements for a 3D content creation service (and it should be noted that many other content creation services could be devised, for example the Adobe creative suite itself could be organised to provide such a serviced for the preparation of images and videos), a high level collaborative framework can be developed. This user interface can be designed with XAML [123], which allows 3D modellers to use existing 3D authoring tools such as 3ds Max or Blender for example, and, through their scripting engines integrate the eMove-chat application to allow collaborative communication between 3D modellers and museum

experts. The 3D modellers would work on separate elements of the 3D reconstruction; for example in the scenario outlined above three modellers could work on the Santa Chiara Church, the Chapel and the High Altar and merge these 3D elements into a common network view for consideration and suggested modification by the museum experts. The render grid (see Chapter 5) and other functionalities would be accessed through a set of web services organised within a workflow management system for rendering frames for a 3D movie style walk-through of the final result. All digital content created in the interface is exchanged with other components using COLLADA, which is an open digital-asset exchange schema [124]. Most of the famous 3D content creation authoring tools such as 3ds Max, Maya, and other tools such as Photoshop and so on supports COLLADA. One of the main advantages for adopting COLLADA in DISPLAYS is that it defines an XML database schema for the data entered, also it is extensible, and hence it can be extended to meet any future requirements.

Other examples of a content creation service could be to develop a museum virtual reality editor (MuseumVR) for museum curators to provide a visual tool for creating virtual exhibitions. A very nice example of such an approach is NeuroVR [125], an open source application that incidentally adopts the same open source code from Blender [95] to create a 3D editing environment. Another service that could be adopted inside DISPLAYS is the EPOCH ARC 3D web service [126] that allows you to create 3D models from a sequence of images.

In summary, the innovative concept in this approach enables the development of a collaborative interface using XAML to integrate specifically designed, commercial (e.g. Adobe tools) and open source content creation and authoring tools to establish a new collaborative content creation environment for digital heritage use. Critical to such an approach is the adoption of standards, such as COLLADA export and import utilities, which can be developed to enable conversion of content between two different 3D environments, and connection through a distributed network or a virtual private network environment to provide collaboration and rendering services to the users. This should ideally be managed through a service orientation approach (web services) designed as a part of a well managed workflow system.

3.2.2 DCA Services

An efficient archival system is important for any digital library to actually archive the digitized heritage objects that have been created using any content creation service as part of the DISPLAYS framework. More detailed information regarding the requirements for archival services is published in [117]. As the DISPLAYS system will also be dealing with different digital formats (audio, video, text, 2D and 3D object and models), there is a need for a robust archiving system that can handle the archiving, preservation, retrieval and editing requests of digital heritage objects. This archive needs to store heritage data and information across the application architecture (i.e. any particular implementation of DISPLAYS including a distributed implementation exploiting social networking, for example), including associated metadata, descriptions, and multimedia content. DISPLAYS therefore has to offer services to deal with different repositories stored in different types of databases, possibly including backwards compatibility with previous solutions (i.e. ARCO), but definitely third party solutions such as Facebook, YouTube and Flickr social network repositories that may be exploited through their provided web services. This archival system can then be accessed by different users of the system to interpret and use as necessary.

As mentioned previously, ARCO has been successfully completed and is now commercially exploited; it allows the museums to build their own database and manage digital heritage objects within it. ARCO also enables museums to build virtual exhibitions or virtual online museums. However ARCO is primarily designed to be installed in a museum along with all of its components such as the ARCO Content Management Application (ACMA), XVRML web server, and augmented reality interface (ARIF), etc. The web contents (digital heritage objects) from the ACMA Cultural Object Manager are transferred to an ARCO virtual museum web server (XVRML Server) via organisation in the ARCO Presentation Manager, which is then integrated into a museum's website in some way (i.e. dynamic content in the form of digital heritage objects are integrated into static XHTML content to form an online exhibition or virtual museum). However, ACMA is a local or remote desktop application, which, although a tool that can be used for providing archival and presentation services, is not open source and it is the whole tool itself (ACMA) that is

linked to the database is a web service. It is also intended to be used directly by heritage, archaeology and other related management communities, not the public community. A critical concept of DISPLAYS is to extend the archival and presentation services of 'ARCO type' systems to the public through the exploitation of Web 2.0 and social networking technologies.

Although the content inside ARCO is usually managed by the museum officers, the DISPLAYS concept allows the public to participate and share their 'finds' or personal objects in their possession with museums and other communities of interest. If it is required to give a management access to a general user and also involves a huge security risk, this task looks quite complex. However, as indicated above, this requirement can be fulfilled by involving some public and commonly used applications such as Facebook, Flickr and YouTube as repositories, and integrating these repositories as a part of a DISPLAYS archival system. This ability, the integration of social networking to act as a community repository integrated with the museum repository, has now been implemented as part of the RCH digital repository, mentioned in Chapter 1 and above. Through this kind of a system, public users can archive their user-generated content (digital heritage objects) on public repositories, linking these with associated museum objects, and the data from these public repositories can easily be retrieved using their open source provided web services Figure 3.6 illustrates two example DISPLAYS archival services architectures.

Starting with content creation, a digital heritage object can be archived by exploiting the ARCO system for storing the objects and ACMA for retrieving the archived contents. The contents from ARCO can be exported on an exhibition server for display using XVRML templates already designed and available inside the system. The ARCO administrators can also design their own templates and use these for building their own expositions (or exhibitions) [127].

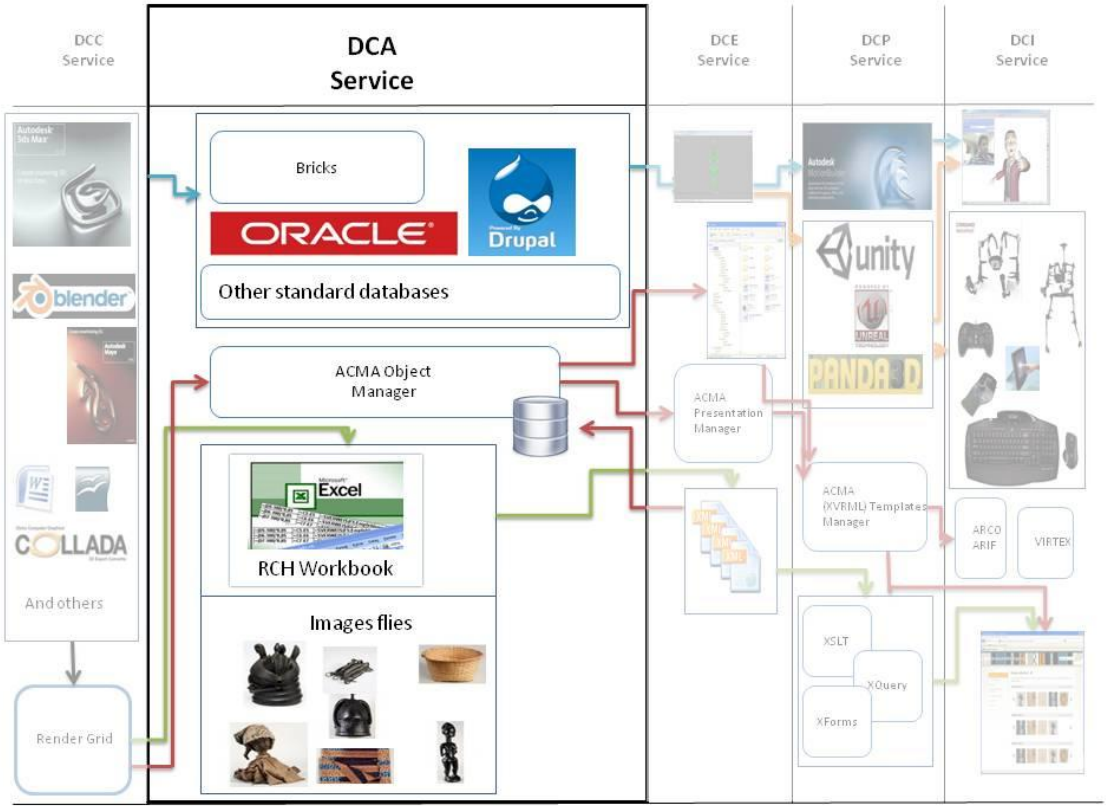

Figure 3.6: DISPLAYS Services diagram with highlighted DCA service

An alternative archival method, named the Lite archival system in Figure 3.6, which is a much faster method of accomplishing archival and retrieval tasks, is to allow a user to store their digital heritage object data in a set of tables defined in an Excel workbook and then use the associated tables to generate XML files that can be interpreted allowing any digital heritage objects embedded to be presented inside an exhibition in two ways:

1. The Excel-based data (i.e. the XML files) can be stored in the ARCO database using the import function inside ACMA. A DISPLAYS archival conversion tool converts the Excel work sheets to ARCO XML format, these digital objects can then be used from ARCO as they are already inside the system.
2. Alternatively the Excel worksheets can be exported directly to an XML format so that the exhibition server can present the object for display using XSLT, HTML and CSS (cascading style sheets). I.e. the dynamic digital heritage objects formatted in XML are integrated with the static content of the virtual museum.

The DISPLAYS Lite archival functionality is really useful in cases where some users would not require training to learn the whole ACMA tool but they can simply use a predefined Excel worksheet to perform the similar operations and archive or export them easily. Also, this can be useful where either ACMA cannot be installed (i.e. the system with less privilege access or where the people would not bother to install new software applications for small amounts of data), or the locations where ARCO servers cannot be reached (i.e. computers with no or poor Internet connectivity, thus providing a portable presentation and data input method).

3.2.3 DCE Services

Digital heritage libraries cannot be complete without appropriate services or tools for creating exhibitions or display environments where museum curators can display their digital heritage objects in context with other objects, and a story or visual narrative that explains what these objects are. The same applies for physical heritage objects, whereby a museum puts on an exhibition. We propose, therefore, that DISPLAYS exposition services provide similar exhibition environments, but in the digital world (e.g. a virtual museum).

The idea behind a DISPLAYS exposition service is that the information archived on different repositories can be presented in some sort of understandable format, e.g. as a website containing multimedia (including 3D) representations of museum objects (i.e. digital heritage resources and objects). In other words our proposed exposition service is aimed at creating many virtual exhibitions for already archived digital heritage objects.

We have established that a goal for an exposition service is such that a museum curator, or other responsible person, can create virtual exhibitions or museums using that service or tools. A good example of an exposition service is the use of the ARCO presentation

manager[2] with modified XVRML (XML Virtual Reality Modelling Language) templates to build a first prototype of a DISPLAYS exposition service for the RCH digital repository.

One of the reasons for using the ARCO presentation manager as a proof of concept for a DISPLAYS exposition service in the RCH repository is that ARCO itself was a successful system for building virtual museum and expositions [127]; as such this is still a valid solution for building a DISPLAYS type of exposition service with some modifications. Using the ARCO presentation manager as an exposition service allows digital object data and relevant metadata to be organized in a structured way in exhibition spaces, where each space is called a separate expo or exhibition, with associated digital heritage objects. The layout for visualisation and interaction domain in these expos is controlled by XVRML templates: an XVRML template is based on the XVRML language developed by the Poznan University of Economics; it is basically an XML wrapper language around the VRML scene graph language to allow loops around dynamic 3D (and multimedia content) and connection to the ARCO database. A virtual museum exposition is composed of a series of exhibitions spaces, where each of these exhibitions is designed with an XVRML template that structures the layout of heritage data and information into a web-based virtual exposition. This exhibition can then easily be published on the web through a DCP (Digital Content Presentation) service, e.g. a web server such as Apache Tomcat [128], that can understand XVRML templates and produce client understandable format to display the outcome on the user's computer.

The contents in a digital heritage exhibition (e.g. virtual museum) are called digital heritage resources, since in the proposed DISPLAYS framework we deal with two types of heritage resources:

---

[2] In this case really a tool rather than a service, because the presentation manager is a tool within ACMA, which itself is configured as a desktop application connected to the database via a web service. It is rather unfortunate that in ARCO it was named Presentation Manger rather than Exhibition or Exposition Manager).

1. Dynamic digital heritage resources
2. Static digital heritage resources

It is critical that the information available in both formats should be reasonably sustainable, secure and robust to future degradation of the exhibition over a reasonable period of time after any particular exhibition is no longer supported. Therefore, our proposed exposition service also suggests the implementation of a backup and recovery service for both static and dynamic heritage resources. For this LOCKSS (Lots Of Copies to Keep Stuff Safe) [129] could be used to create automatic backups of an exhibition (essentially a web site) on different machines registered with the LOCKSS system, and in event of any failure, one of the backup machine with the latest backup files takes place as a replacement. However, LOCKSS is not specifically designed to safeguard web sites in this manner. There is a suggested method on how to modify LOCKSS to achieve this solution [129], which is beyond the scope of this thesis.

Figure 3.7 describes an example workflow from archival to exposition services with all major components in both areas. The administrator or museum curators of the system can create many different exhibitions (with a DISPLAYS system implemented with the described ARCO components), which will be stored as web pages on the exhibition server. The museum web server closely works with exhibition server (in some cases it can be a single physical server machine) to display the exhibitions on the Internet or even through a museum kiosk system. A DISPLAYS exposition server is also expected to integrate with other third party web services, e.g. through Web 2.0 mashups and social networking services to provide easy access to end users. The diagram also shows LOCKSS for backup and recovery. This automatically makes many copies of web and exhibition servers on six different machines, and recovers them when there is any fault in any of the server machines.

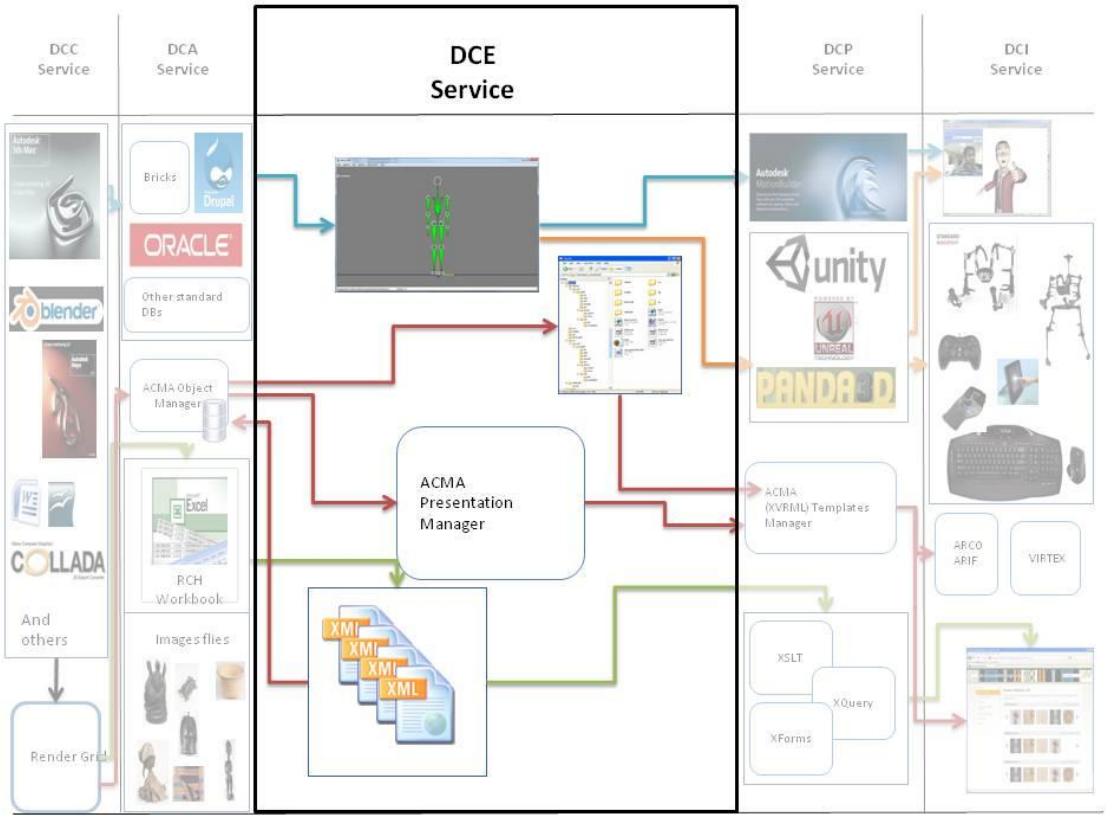

Figure 3.7: DISPLAYS Services diagram with highlighted DCE service

Note for Figure 3.7: in the Lite implementation of a DISPLAYS archival and presentation service, the method of creating the exposition is to embed the 'heritage information' or collection of objects into an XML file, which is then dynamically presented via a DCP service (as a virtual museum for example) using CSS and XML technologies such as XSLT or XQuery. Different exhibitions requiring different XSLT style sheets and CSS; therefore, the XSLT and CSS module is shown under the DCP service. However, this is not clear-cut, because it is difficult to separate 'exposition' from 'presentation' using XML (and XSLT), because the XSL template code also determines the exposition organisation and layout to some extent. Nevertheless, we choose to place the XSLT technology in the DCP service for two reasons: 1) the embedded XHTML code that resides in the XSLT style sheet can be thought of as part of the presentation service, which can be handled by any number of web server technologies, and 2) the XSLT process is either embedded in the browser technology or installed as a DCP service.

3.2.4 DCP Services

As mentioned above, the digital library presentation service is concerned with look and feel (defined by the static XHTML code and its position on a web page, including modules where dynamic content is integrated and which presents digital heritage objects as final output to the end user – in a virtual museum for example). The DISPLAYS presentation service is primarily about the integration of Web 2.0 technologies that combine the dynamic contents from a DISPLAYS database implementation (e.g. the ARCO database or a flat file XML repository, etc.) with other third party web services (that contain user generated contents) such as social networks: Flickr, Facebook, YouTube, Yahoo and Microsoft Bing search engines. A specific example is the RCH digital repository where a user can search for museum content in association with user generated content in the same web interface (mashup). Figure 3.9 illustrates the Web 2.0 mashup of several search engines using the respective social network web services (i.e. their APIs exposed as web services). The presentation environment also provides different ways to interact with 3D objects (discussed in more detail in the next section). Figure 3.8 highlights DISPLAYS conceptual framework for DCP services with many tested software components listed.

The proposed DISPLAYS presentation service uses many Web 2.0 technologies and their respective web services to implement desired functionalities of the presentation service. The major technologies included are classified into Web 2.0 mashups and social networking. These technologies are described as:

1. A mashup is an application or a webpage that combines the data or services from different sources and produces a new application or a website [130]. Data mashups from different public and private resources are used to a new presentation service.

2. A social network is an online application, i.e. a website, where the users create their own profile and build their own social network. This allows users to connect with their old friends, find new friends or connect with people having the same interest [131]. There are many online social networking sites available

free of cost: this is advantageous for the building of community networks for several reasons, not least the ability to manage user groups and exploit storage capacity on public repositories. For example, in the case of the RCH digital repository several APIs have been used (Facebook, Flickr, YouTube, and so on). These APIs use web services (in general most use REST based web services) to expose the API functionalities to build a community network of people with a common interest, e.g. heritage interests.

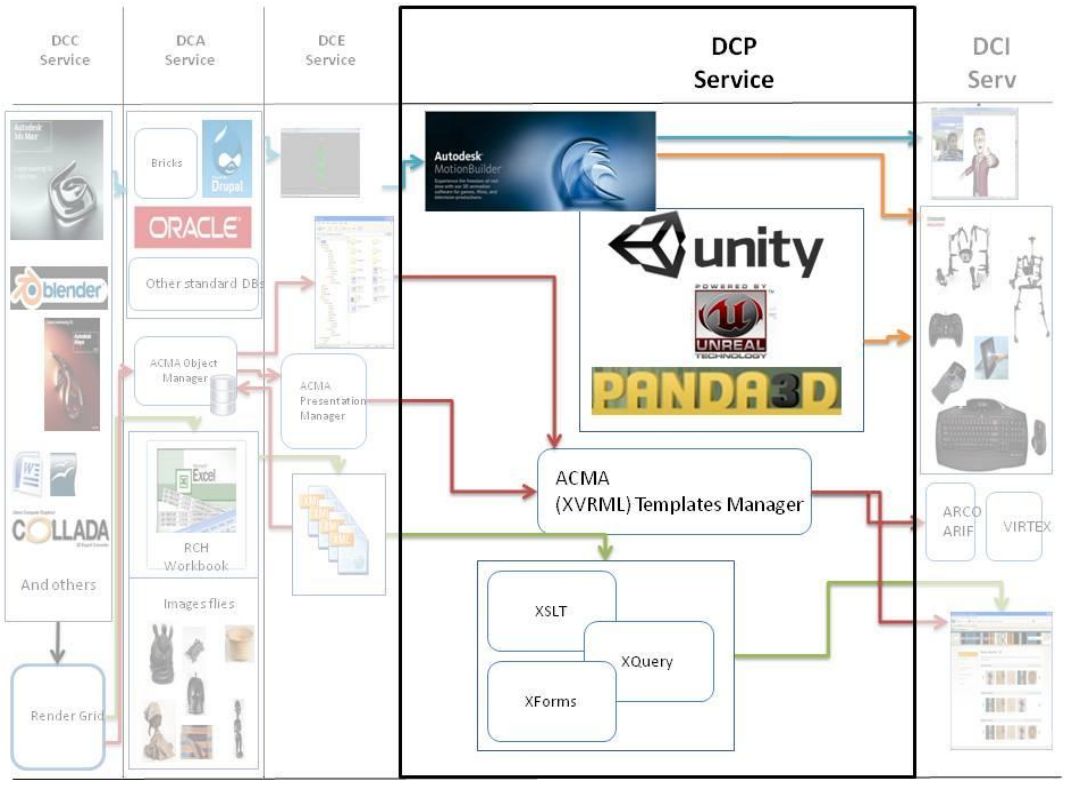

Figure 3.8: DISPLAYS Services diagram with highlighted DCP service

As described previously, DISPLAYS is a theoretical concept with many of the concepts proved in practical examples. Figure 3.9 illustrates an example where a search facility uses web services to access the ARCO data repository for the museum content, and several social networking repositories for presenting user-generated content to the user. In this example, the user (as an example of an interaction service – and this also illustrates how close presentation and interaction services are integrated) can drag and drop images of the objects returned onto a 3D VRML scenes for further browsing and saving for future use. Specifically, the user selects a radio button to access the search

result; the user can then dynamically set up the VRML scene to accept a returned digital object result.

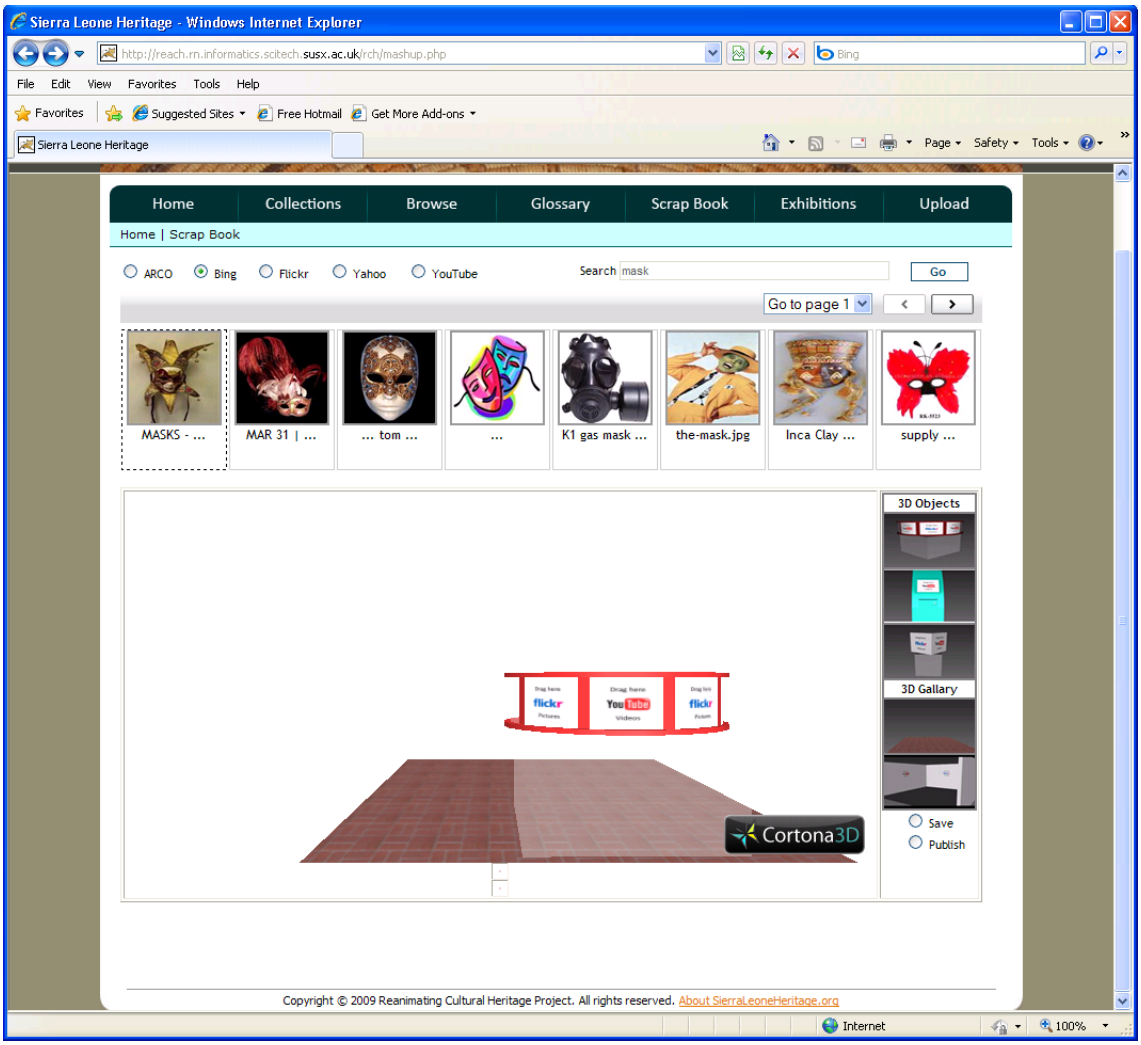

Figure 3.9: Reanimating Cultural Heritage digital repository presentation

Figure 3.10 illustrates another presentation service where a reanimating cultural heritage community on Facebook is integrated inside reanimating cultural heritage website. In this case, the Facebook application as a whole is embedded in an iFrame at the time of implementation purely to test the concept. As an aside, Facebook have currently stopped this practice for understandable security reasons based around login to Facebook has to occur on their site or using the specific Facebook login API. This is another practical implementation of the social networking concept from DISPLAYS framework.

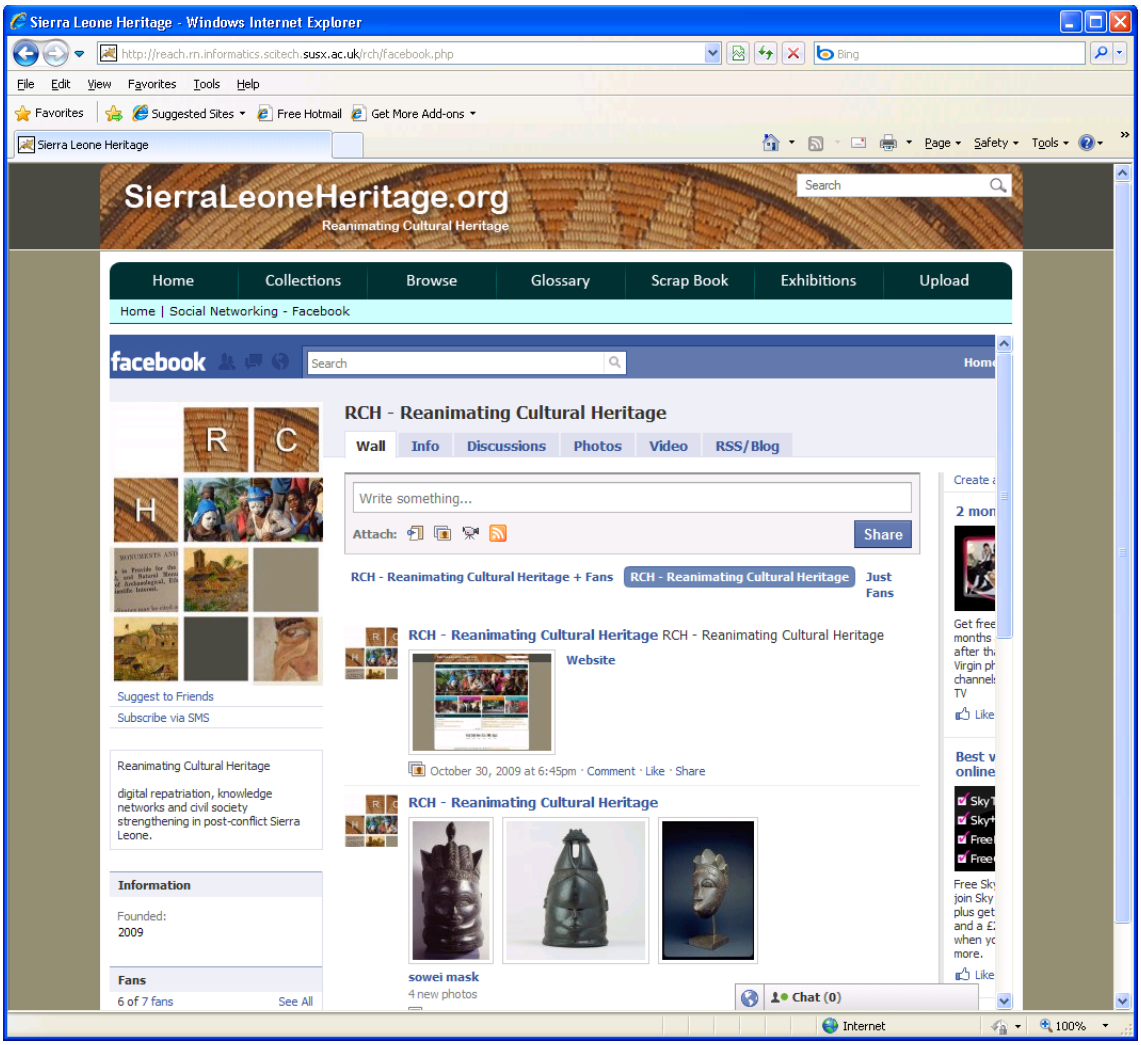

Figure 3.10: Re-animating Cultural heritage community on Facebook

3.2.5 DCI Services

The DISPLAYS framework proposed places no limitations on the variety of interaction services that can be made. Many interaction services have been mentioned above, including the Augmented Reality Interface (ARIF) that is integrated in the ARCO system. Such interaction services can include the implementation of physical hardware devices that a user may use to effect an interaction with the digital heritage resource or object. An example here is the VIRTEX system, whereby a physical replica of a museum artefact is created with associated electronics and sensors that the user handles in a tactile manner to unfold the heritage story of the object [132] Other interaction services may be purely software solutions such as the 3D reconstruction of the Santa Chiara Church implemented as a touch screen interactive [133].

For practical reasons it is likely that interaction services involving hardware developments (for example the implementation of a touch screen interactive, or the use of a motion capture suit) will be located in museums, libraries and other central locations due to their cost factor and purely software interaction services that are accessible over the Internet for both home and museum users. Below is the description for both software and hardware interaction types proposed in the DISPLAYS framework. In addition, Chapter 5 is dedicated to a novel hardware interaction system based on the development of a real-time motion capture technology to present a digital historical character that can narrate a heritage story, which also includes eMove-chat as part of the software interaction. Figure 3.11 illustrates the types of interaction services envisaged in a DISPLAYS framework, these are however not exhaustive.

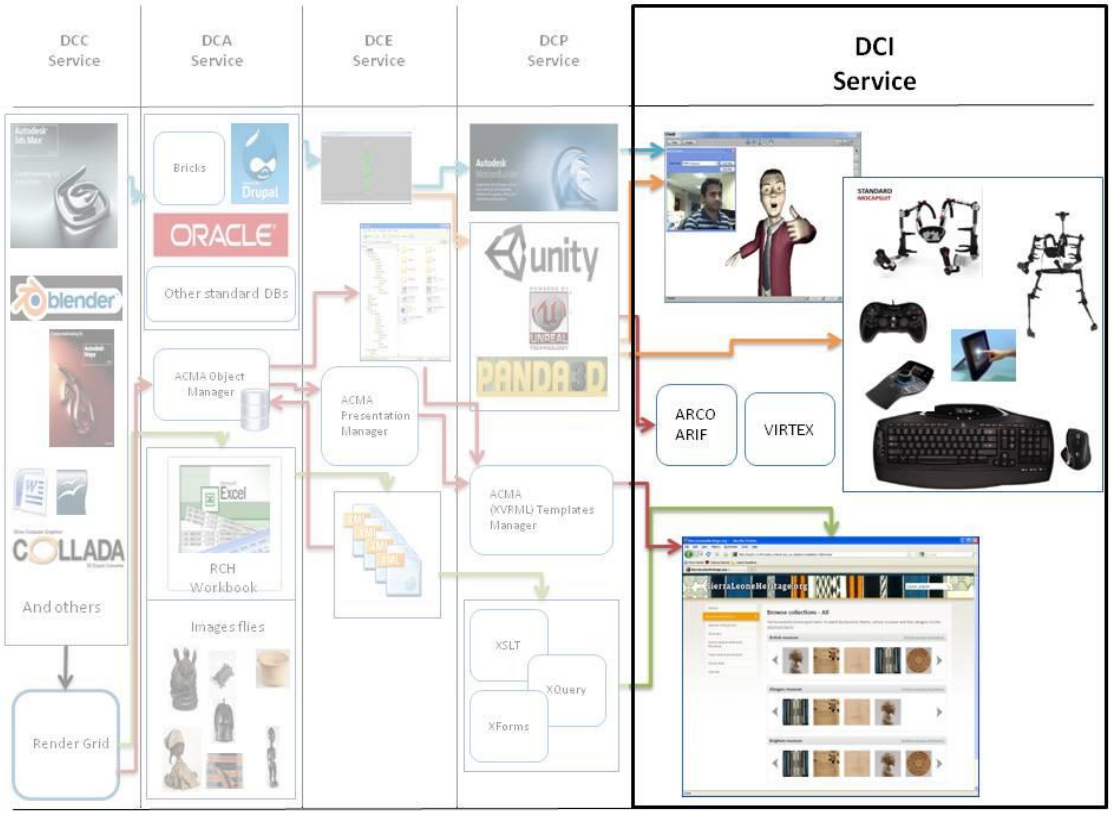

Figure 3.11: DISPLAYS Services diagram with highlighted DCP service

3.2.5.1 *Software Interaction*

Having created any particular virtual museum exposition and presentation service, interaction services can be created in varied and interesting ways by exploiting Web3D

technologies such as virtual and augmented realities and other multimedia effects. This type of visualisation and interaction has already been demonstrated in the ARCO project, using Virtual Reality Modelling Language (VRML) [134] as part of the ARIF component. Therefore, ARCO functionalities can be used as a base for further development of virtual and augmented reality interaction services to enhance these methods in the DISPLAYS framework.

One of the examples from DISPLAYS software interaction can better be explained through Figure 3.9 already described above. Here the user can select a heritage environment in VRML, then search for a heritage object from any of the available services (e.g. ARCO, YouTube, Flickr, Bing, or Yahoo), and drag and drop the searched objects inside the heritage environment on their VRML object place holder. As shown in Figure 3.9, some simple VRML objects are shown coded with movie and image textures to accept drag and drop cultural objects. The user can also move, rotate, zoom in and zoom out these VRML objects and on completion of the newly created environment they can also save this environment on their own computer. This combined presentation and interaction service is done in a web browser from any remote location and the user has to install nothing except a Cortona3D plug-in [135] inside their web browser to allow VRML files.

Additionally, DISPLAYS proposes to use the eMove-chat (voice, video and text chat) facility for conversations between people at different stages from development to collaboration, data or information sharing, and heritage interaction service, i.e. a remote version of digital puppetry for digital heritage. The eMove-chat facility is discussed fully in Chapter 5 of this thesis.

3.2.5.2 *Hardware Interactions*

The hardware interaction methods proposed in the DISPLAYS framework, as mentioned can be many and varied, but includes as a notable example the VIRTEX [132] interface (i.e. replicas of 3D digital heritage objects used to control a virtual object in a virtual heritage environment).More importantly, in the context of this thesis a proposal for a digital ‘Interaction through motion capture’ to unfold a heritage story is presented in Chapter 5. This system exploits the eMove exo-skeleton motion capture

suit [http://www.mocapsuit.com/] to interact with digital heritage objects and navigate through reconstructed digital heritage environments.

3.3 DISPLAYS Scenarios

Up to now a description of the DISPLAYS concept or framework has been given. This section briefly describes some of the possible and important scenarios for a DISPLAYS type of architecture. These scenarios will briefly explain how DISPLAYS can benefit the community and society, and how the overall system works in terms of individual components.

3.3.1 A Scenario for Portable Antiquity

The following scenario is based around the archival of portable antiquity type heritage resources inside a DISPLAYS archival, exposition and presentation system.

A local rambling society is walking across the countryside and one of them discovers an artefact that looks interesting, old and has certainly been created by man, e.g. a piece of pottery, a precious stone, the stem of a clay pipe or any other valuable heritage resource. The member of the rambling society informs the rambling officer who organised that particular ramble about the artefact. The rambling officer notes the location (takes a photograph of the site) and makes a short description about the artefact on his note pad in his iPhone, and also the GPS co-ordinates he can access on his iPhone. After returning home, he accesses the DISPLAYS archival service using the interface provided on the rambler's society web site to record the artefact. Several members of the local metal detector society who are also member of the DISPLAYS heritage library have also discovered portable antiquities in the same location, and they also use the same archival service to record their findings including location, video, images, and other metadata elements agreed with the local archaeological officer. The metal detectorists work closely with local archaeologists, so they have a trusted log-in to the archival service with the archaeologist, and they can also use the same archival service to access and store their findings. Note: the location information about the objects is restricted to selected members and not allowed for general public access. This is to

prevent ‘treasure hunters’, who cannot access the location data, from locating other possible treasure in same area.

Any new data submitted is archived in a separate folder inside the ARCO database object manager (or a specified Excel worksheet if the DISPLAYS Lite system is used) and the administrator of the DISPLAYS system is sent an automatic email with notification about the new object. A DISPLAYS; system administrator (i.e. a museum officer) verifies the newly entered object before inclusion in any future exposition.

3.3.2 Collaboration in Content Creation

The proposed DISPLAYS architecture also offers collaboration between 3D modellers in content creation, i.e. 3D model design. The following scenario justifies the collaboration and high performance computing environment in DISPLAYS.

The Sussex Archaeological Society has been granted limited funds to spend on public engagement focused on their Fishbourne Roman Palace archaeological site. The site itself is well-known, and has many thousands of visitors per year including school groups. However, the CEO (Chief Executive Officer) of the Sussex Archaeological Society feels that greater public awareness and access could be achieved with an online virtual museum including a 3D reconstruction of Fishbourne Roman Palace. This could perhaps be implemented as a touch screen interactive, similar to the one recently installed by the University of Sussex Computer Graphics Centre, in the Victoria and Albert Museum (a 3D reconstruction of the Church of Santa Chiara in the Medieval and Renaissance Galleries).

Fishbourne Roman Palace is very large, perhaps the largest Roman Villa in England, and as such is referred to as a Palace rather than a Villa. As such it has many parts, and after discussions with the Sussex Archaeological Society, the Computer Graphics Centre realised they could implement the required 3D reconstruction within a DISPLAYS exposition, presentation and interaction environment. Furthermore, the 3D reconstruction requires many, many thousands of rendered frames so that an implementation of the render grid is feasible (see Chapter 4). In this context, the Computer Graphics Centre bid for the work and won the contract based on its price.

This was sensitive to the budget indicated and they were able to do this because they could deploy the render grid over both the Sussex Archaeological Society and University of Sussex computing infrastructure.

In addition, the bid included the notion of developing the 3D reconstruction as part of a team-based Master's project, and as such the 3D reconstruction was divided up across several digital heritage Masters students. To assist in collaborative development, and in order to include the Sussex Archaeological Society museum experts, a DISPLAYS 'collaborative 3D digital content creation service exploiting the render grid' is set up, see Figure 3.5. Fishbourne Roman Palace can now be created in 3D by many modellers (Master's graphics students are included in this scenario) working in parallel by either designing individual components or using a distributed architecture as proposed and demonstrated [118].

Once the Roman Palace 3D model is finished, many thousands of 3D frames can be rendered using the render grid service. These rendered 3D images are then ready for integrating into a virtual fly-through of the Roman Palace [113] [114].

3.3.3 Digital Heritage Information Sharing

Similar digital heritage objects (and their physical counterparts) are geographically dispersed across the global museum-scape. A method is required to locate these digital objects within the same digital space for access by both the museum expert and other communities of interest. Implementing DISPLAYS archival, exposition, presentation and interaction services can allow such digital objects to be shared among the relevant and interested communities for their mutual interests. Web 2.0 technologies, such as mashups and social networking in DISPLAYS, can be used to achieve this goal.

A specific example of this scenario is achieved in the RCH digital repository, whereby individual museums such as the Glasgow Museum, the British Museum and Brighton Museum and Art Gallery archive their contents on an archival system (using DISPLAYS with an integrated ARCO database or the DISPLAYS Lite XML system), which is then exposed to the web using ARCO or the Lite exposition services. The specific DISPLAYS validation architecture, i.e. the RCH digital repository, does the

same but uses an Excel workbook system to manage the digital objects and an XML repository for archiving, with CSS and XSLT to organise the exposition, with standard web server technologies (e.g. PHP, etc.) to present the collection as a virtual museum online. Web 2.0 mashups that integrate YouTube, Flickr, Bing, Facebook services, etc., so that user generated data from the community can be integrated with museum content, is also included. Of particular interest is the inclusion of social networking technologies such as Facebook, MySpace, and Orkut (RCH uses Facebook) to allow communities of interest to share their interest on a single platform. Individual community users can build their own sub-communities and share digital heritage resources and objects that are already in the digital heritage repository.

3.3.4 Creating a Virtual Exhibition

A museum education officer, who has an idea for presenting a virtual museum exhibition based on their museums heritage collections, wishes to create a virtual museum for presenting Sierra Leone artefacts held in the museum. The education officer accesses the museum's collections management system to access any digital records of Sierra Leone artefacts they have. The search on the museum catalogue discovers some 700 artefacts that are digitised with appropriate metadata, images and textual descriptions. The museum education officer simply dumps the collection into an Excel worksheet format that has been provided by the DISPLAYS system. Static XHTML code is developed by the museum's IT department, with appropriate 'modules' (e.g. using PHP, or any other server or client scripting technology) for receiving dynamic digital objects from the collection. The data is simply exported as an XML file from the Excel worksheets.

In this way a virtual museum exhibition can easily be created using the information archived in an XML repository. Alternatively, the museum education officer uses the DISPLAYS exposition service, which is actually a modified ARCO exhibition service, to create virtual museum files using available XVRML templates. The DISPLAYS exposition service points to the web publishing folder where these exhibition files are stored on completion. The museum officer then links the exhibition URL inside their heritage website to publish its contents on the World Wide Web.

3.3.5 Real-time Motion Capture based Historical Character

This particular scenario can be seen in detail in Chapter 5 because it is the scenario that justifies the innovative implementation of the 'Interaction through motion capture'.

3.4 A DISPLAYS System Architecture

The DISPLAYS framework is concerned with creating tools with a service-oriented approach for the implementation of a digital library system (digital heritage system). The overall architecture proposes to use cutting edge tools and technologies to build any specific implementation of a whole system or components of a system. DISPLAYS is focused on allowing heritage amateurs and professionals to create, archive, use and explore digital heritage content with other people in interesting visualization and interaction environments using Web 2.0 and various innovative hardware technologies.

Further to this, DISPLAYS can include the adoption of various game engines as a part of a digital library presentation and interaction service to provide hardware and software interaction with 3D digital heritage environments and objects (useful for the implementation of motion capture methods for real-time avatars – heritage characters). Current implementations (RCH repository and other components) have proved the viability, for example, of using three gaming engines, namely Panda3D, Unity3D and Unreal, to connect a real-time motion capture suit to these gaming engines. This allows the implementation of the heritage character scenario, where an avatar is driven in real-time by a person wearing a motion capture suit who is enacting the persona of the heritage character to unfold the heritage story. A brief discussion on these tools can be found in Chapter 2, and more specific details on the innovative implementation can found in Chapter 5.

The DISPLAYS framework proposes a heritage data model based on the integration of 'data', 'information', 'knowledge', 'display' and 'play' components implied by corresponding services described above in Section 3.1 – these services or components are the backbone of the system. By using a service oriented approach as an overall architecture, it is envisaged that each element of the heritage data model can be accessed through some set of web services, allowing more portability and robustness for

the overall system. The individual services can be deployed in different places where required, and, similarly, a complete workflow management system can hold these individual services as a system for museums and other participants of a heritage library.

3.4.1 Service Orientation

Service orientation allows us to create independent services portable on different types of computer systems to build collaborative applications. This allows the user (i.e. museums and communities) to aggregate heritage data and information from different sources and data repositories. Similarly, it allows the aggregation of DISPLAYS based digital heritage objects with several public social networks such as YouTube and Flickr. The render grid services can also be placed on different locations to perform computer-intensive operations and use idle CPU cycles in the institution (museum, etc.). At the same time the same service orientation allows us to build a user-friendly management interface for end users to access the system. The DISPLAYS prototype application (reanimating cultural heritage) is also implemented using service orientation based on a set of services (i.e. DCC, DCA, DCE, DCP and DCI services) that communicate with each other through predefined workflows to implement all the digital heritage resource (DHR) functionalities.

All the major functionalities inside DISPLAYS are deployed in a service-oriented architecture, such as in the ‘reanimating cultural heritage’ project. Different web services are used to perform specific operations and return heritage results. Figure 3.12 below illustrates the service oriented architecture for reanimating cultural heritage project; this is currently in its implementation stage.

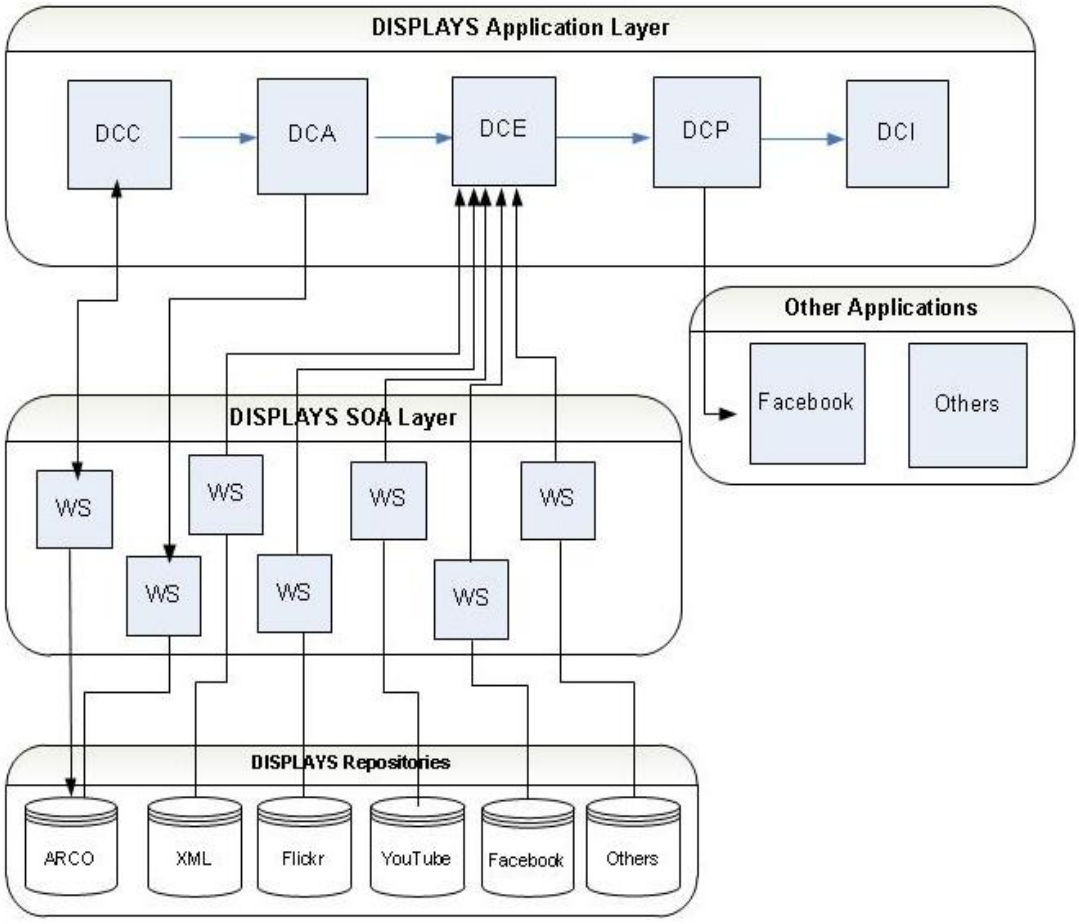

Figure 3.12: DISPLAYS Modified Service Oriented Architecture for Reanimating Cultural Heritage

These services implied in Figure 3.12 are used to glue together the whole architecture as a DISPLAYS system. System expansion and portability are the key benefits of service orientation; notice all the data repositories in DISPLAYS, as illustrated in the reanimating cultural heritage system, are accessed through service orientation. This allows us to expand the system easily, integrate more repositories or replace existing repositories with alternative solutions without an extremely long and painful process.

Third party applications, such as Facebook, YouTube and Flickr, provide significant support to integrate their APIs (exposed as web services) into existing systems, and many applications on the Internet benefit from these services to implement their own innovative ideas. DISPLAYS uses these web services to extend its functionalities; for example, it uses Flickr to share community generated content that describes digital

objects through images, and Facebook to collaborate with peers in the digital heritage community.

Figure 3.13 illustrates the DISPLAYS service orientation architecture with the DCI services expanded where the hardware tools such as the eMove exo-skeleton suit, VIRTEX interface, and some software environments such as 3D game engines and eMove-chat applications, are suggested and tested for implementation.

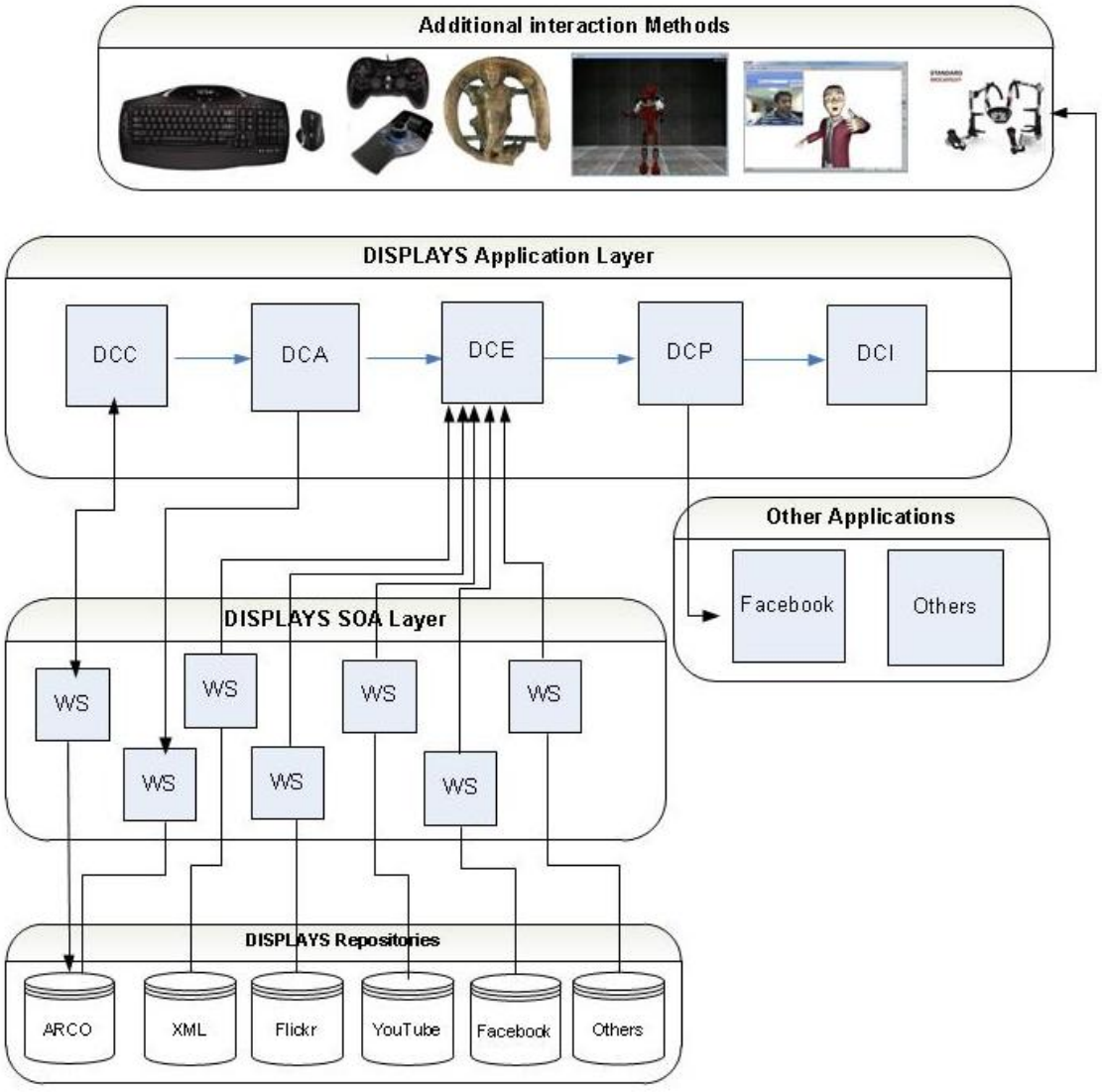

Figure 3.13: DISPLAYS Service Oriented Architecture with additional interaction layer

3.4.2 Workflow Management

The workflow management system is about creating an automatic system management process that contains predefined procedures to deal with user requirements and these procedures are constantly executed. DISPLAYS is a conceptual digital heritage system consisting of a system architecture and data model that can be complex, and may require different components and services to operate together, either sequentially or simultaneously, within a particular hosting environment in many use case scenarios. The workflow management system is refined in such a way that all unnecessary steps are removed, making whole system work faster and more efficiently.

As DISPLAYS is a framework concept with many individual implementations, the complete workflow managements system is beyond the scope of this thesis in terms of any particular implementation. However, some details of the scope of a workflow management system can be used to control many aspects of a DISPLAYS architecture, e.g. a workflow management system could be implemented in Python to allow communication between the user and the render grid (see Chapter 4) to allow the collaborative modellers of the system to use available computation power to render high quality images and animations [113].

The DISPLAYS system should be built on a distributed infrastructure that enables the system to be flexible and suitable for sharing heritage knowledge in the form of archived data and information. These are transferred over the network throughout the system; this requires state-of-the-art workflow management services and techniques. The effective workflow to such a system is vital in order to increase the efficiency in transferring and accessing the available information by users of the system.

DISPLAYS proposes different workflow management services for different service areas (DCC, DCA, DCE, DCP and DCI). These workflow management services should be designed to manage the flow of different heritage resources that will be stored in the system, from resource acquiring to the point of retrieving and displaying them to the user in a visualisation and interaction scenario. The workflow system will, therefore, be used to automate this process. In short, the workflow management system will handle

the task of coordination and data flow between applications. Below are examples of some of the possible workflow management systems for the DISPLAYS framework.

Workflow management tools for:

1. Render grid
2. Mashups and social network technologies
3. Digital content creation
4. Archival of 3D contents / objects
5. Virtual exposition
6. Accessing the presentation services
7. Interaction with 3D virtual objects using keyboard and mouse
8. Interaction with 3D virtual objects using motion capture suit
9. Digital puppetry
10. Backup and recovery using LOCKSS

There are two important characteristics that are considered here while designing the heritage workflow system:

1. Integration of the system components
2. Interoperability between application and infrastructure components.

Specific heritage workflows provide the system with the ability to distribute tasks and information between components of the system and users, or between two components or two users in the system. This may involve the process of sequencing various activities that are a part of a particular process, e.g. building an exposition such as a virtual museum, etc. This thesis limits itself more to a brief discussions on the potential for workflow management of the DISPLAYS system, further, the concept of workflows in the DISPLAYS architecture is discussed in other published work [7] [117].

3.4.3 Summary

This chapter presented a detailed description of the DISPLAYS conceptual framework, and its applicability for implementing digital heritage systems through various scenarios. DISPLAYS is a proposed framework for digital heritage library systems,

where different methods and techniques using cutting edge technologies are presented. DISPLAYS services include: DCC (Digital Content Creation), DCA (Digital library Content Archival), DCP (Digital Content Presentation), DCE (Digital Content Exposition), and DCI (Digital Content Interaction) services. DISPLAYS proposes a community based collaboration in content sharing using social networking repositories such as Facebook, Flickr, YouTube, etc. The DISPLAYS system architecture is designed with service orientation in mind for easy integration and future expansions. It also proposes a workflow management system for each component, and an overall workflow management system for DISPLAYS types of digital heritage systems.

The focus of this thesis is to produce innovative results for DCC and DCI services, and present these results in a DISPLAYS type of architecture to produce unique grid-based rendering and motion capture based interaction services.

CHAPTER IV

4 Grid based Content Rendering

4.1 Introduction

The growing demand for scientific simulations and their representations that visualise results through 3D graphics requires high quality 3D models and animations. Designing high quality 3D models (e.g. models with more complex geometry and photo-realistic rendering) not only requires more time for design and modelling but also requires more time and computational power for its rendering.

There is no doubt that rendering takes a huge amount of time and causes delay in deliverables on many occasions. The rendering process assumes time and power are inversely proportional to each other, i.e. the less computational power you have the more time you require to render and *vice versa*. Obviously time is more crucial for professional projects development and deliverables on time. Hence, different kinds of render farms have been proposed by many artists, developers and graphic communities to reduce the overall time cost involved in computer graphics rendering.

Photo-realistic and powerful 3D applications require greater computational power, forcing organizations to invest money in powerful but relatively inflexible architectures solely dedicated to rendering environments. Traditionally, a render farm requires a cluster of computers to perform satisfactory rendering in terms of computational power and time. Typically, these cluster-based render farms are built around blade architecture [136], which is quite expensive and not every scientific organization can afford it. Ideally a more cost effective solution would be to use the existing machine power

already available inside the organization's control to perform rendering and other computer-intensive tasks.

One solution is a grid based rendering (render grid), where the organisation's existing computers and their network could be utilised to provide the required computational power for rendering graphics applications. The render grid can be built and used for rendering during their off peak usage hours and would be really useful for many organizations, especially organizations with no or limited budget for high performance computers [114]. Also, as it would be a render grid, the machines used in rendering can be used for performing other HPC tasks, such as mathematical expression evaluations, scientific simulations, physics and astronomy experiments, etc.: there could be a long list for HPC applications.

As a part of a DISPLAYS infrastructure framework, various unique solutions have been proposed for the different services that utilizes different cutting-edge technologies. This chapter is focused on one of these proposed solutions: a solution for the implementation of an innovative render grid that provides an authoring tool to render 3D images based on open source grid tools and technologies. This research also proposes to integrate such a solution into wider application architectures, in this case, digital heritage libraries, where digital heritage objects and resources need to be rendered and animated for DISPLAYS applications and scenarios.

Three prototypes of the render grid solution have been proposed for DISPLAYS, and at this stage prototypes 1 and 2 have been completed (i.e. as a standalone Windows application). Prototype 3 is effectively the same as prototype 2; the only difference is that it specifies the service orientation layer of the render grid. As proposed in this thesis, the innovative render grid is designed using Condor; the later versions of Condor are specified to be service oriented. Given that, after the implementation of prototype 2, it was not feasible (or worthwhile) to develop a specific DISPLAYS code to implement the service orientation, rather it is more logical to await the next version of Condor. At the time of writing this thesis a service orientation version of Condor had been released [137], [138]. As such it is not a significant step to implement prototype 3. A

DISPLAYS architect wishing to complete this step would be able to review prototype 2 and the new version of Condor to effect such an implementation.

This chapter focuses on the implementation of the DISPLAYS render grid prototype 1 and prototype 2, with the proposed prototype 3 specification being discussed to show how the final version of a render grid solution should look like in the future. Moreover, a brief discussion about render grid prototype 1 and prototype 2 can be found in work already published [114] [118].

The first prototype was an initial implementation of individual components; it was manually tested to see if everything worked together correctly. During prototype 1, a set of software tools were installed on seven computers (this software included Condor and Blender), and then a render script (grid job submission) was written (hand coded, for later versions the grid job script is generated by software code) based on typical grid job specifications. The newly created job was then submitted to the grid environment using the DOS-based command prompt and the results were retrieved. The purpose of this prototype was to test the functionalities and possibilities of building a render grid using open source grid computing for environment. For greater details on these results see [113].

Prototype 2 focuses on exploiting Blender 3D which is an open source 3D modelling and animation tool, and Condor high through put grid infrastructure [35]; these are tightly coupled via the implementation of a new render grid management tool written specifically for the purpose. This kind of approach is very useful for embedding a render grid solution within a specific application area, such as digital heritage libraries. This innovative coupling of open source Blender 3D (3D modelling and rendering package) and Condor (high throughput computing) with a novel render grid management tool, brings about a unique solution: a 3D modelling, rendering, computational resource to provide a render grid in a very cost effective way, published in [114].

As mentioned above, the final system will move towards a service orientation architecture where the services are offered by the three components (open source Blender 3D, a DISPLAYS render grid management tool, and Condor high throughput

computing environment) and are loosely coupled via web services. The plan also includes making this render grid management tool as a toolbar inside an open source Blender3D user interface. For more details please see Chapter 6 regarding the conclusions and future work.

This chapter is organised as a discussion on a specific render grid scenario inside a DISPLAYS type of infrastructure, discussions on relevant work, system architecture, detailed discussion on each aspect of implementation from grid computing to components of the render grid management tool, unique features of a DISPLAYS render grid with other available solutions, tests and results and is followed by a summarized conclusion of this work.

4.2 Render grid for DISPLAYS

As a part of a digital heritage library framework a set of services are proposed. These services incorporate digital content creation, archival, exposition and presentation functionalities. See Figure 4.1 for digital heritage libraries and render grid usage inside a DISPLAYS infrastructure. The DISPLAYS render grid solution is based on open-source components to create a render grid that utilizes Blender 3D and Condor high throughput computing environment, with a bespoke grid management tool. Such a solution will allow heritage educational professionals, for example in museums, to develop virtual reconstructions, animations of heritage buildings and artefacts cost effectively. The render grid system includes a set of functionalities and unique features, which are not currently available in other solutions. See Table 4.1, below for comparisons of a DISPLAYS render grid with other available solutions.

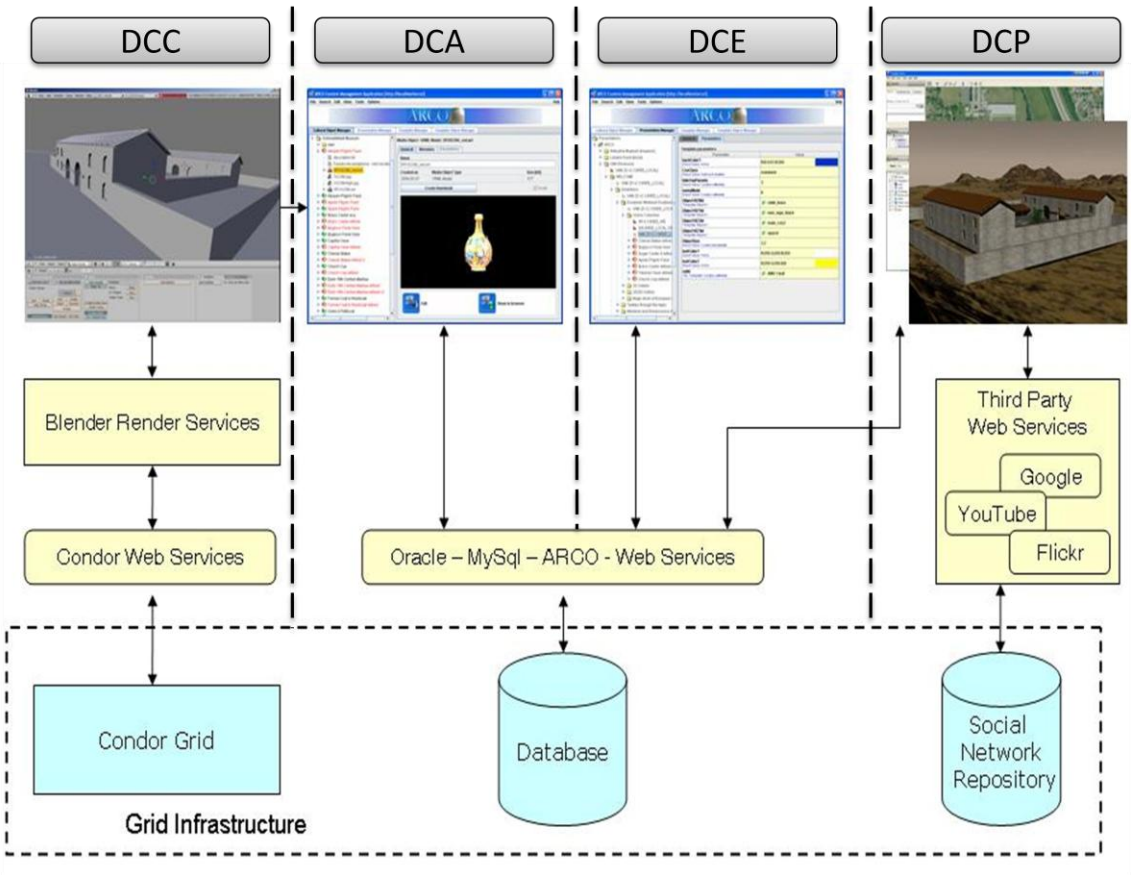

Figure 4.1: DISPLAYS architecture deploying the render grid

Figure 4.1 describes DISPLAYS four services (content creation, archival, exposition and visualization) with a workflow from content creation (Blender3D environment) to an exhibition/presentation service utilizing Condor grid computing. The user of the system can create a 3D object, render it using the DISPLAYS rendering service, archive it using the archival service and finally present it to the community of end users using the DISPLAYS presentation service.

Another aim of digital library services architecture is to develop a 3D collaborative environment based on grid and P2P infrastructure, where the users can collaboratively work on the same 3D model, e.g. virtual reconstructions, animations of heritage buildings and artefacts, etc. In this case, all 3D artists can share the same graphics file and work in parallel to design a common model and animation throughout the day. When they have finished on a stage, they can submit their job(s) for rendering on the grid, eliminating the need for a dedicated render farm server. The 3D collaborative

designer environment is a large piece of work on its own; details on this area can be found in published work [118].

The adoption of a render grid allows the use of idle resources of an organisation's systems when they are free. The user only needs to install a small Condor slave program to be the part of grid, so when the system is idle it is used by the grid master computer as a part of the DISPLAYS grid environment. This is done in the background, and when the user, or any foreground application, is initiated the grid-based application is suspended and allows the foreground application to retake control of all computer resources. This area of research is termed "Harvesting Idle CPU cycles" [139].

4.3 Related Work

As already discussed, rendering is a big issue when discussing 3D computer graphics applications, especially when graphics and animations contain millions of polygons or photo realistic effects. Different solutions have been proposed; discussed below are some well-known and appreciated projects in 3D graphics communities. Later on, Section 4.6 describes various unique features to justify the importance of the DISPLAYS rendering environment in comparison to other solutions.

4.3.1 DrQueue

DrQueue [140] is an open source general-purpose batch processing solution, which is also used for setting up render farms by open source communities. It works on simple TCP/IP connections for executing various kinds of applications on a computing pool (i.e. many computers connected in a networked environment). In addition to various applications, DrQueue is also used for rendering computer graphics applications. DrQueue supports many 3D applications, namely 3Delight, 3ds Max, After Effects, Aqsis, Blender, BMRT, Cinema 4D, Lightwave, Luxrender, Mantra, Maya, Mental Ray, Nuke, Pixie, Shake, Terragen, Turtle, V-Ray and XSI. It also supports many operating systems, including Linux, Mac and Windows. The computing pool used for DrQueue is dedicated for the DrQueue distributed environment only like a cluster of computers, unlike grid computing where these machines can also be used by the end user for their daily routine work. Although a solution based around DrQueue is good for building a dedicated render farm where the organization has money to afford dedicated computers,

this cannot be used for a DISPLAYS rendering solution where the requirement is to utilize idle CPU cycles from an institution's existing available machine power rather than dedicated machines.

4.3.2 FarmerJoe

FarmerJoe is also a render farm implementation based on TCP/IP connections to render Blender 3D models; it provides a user-friendly web based interface to schedule jobs for rendering [141]. Additionally it also supports the functionalities of a Bucket-based technique (a technique in which the frame buffer is subdivided into small regions, which are rendered independently and integrated on completion [142]), and image or frame-based rendering (where the rendering is performed on every frame separately, unlike Bucket-based rendering where every frame is distributed on different machines [143]). FarmerJoe is a widely used rendering solution, but this set up also requires dedicated machines to build a rendering environment (i.e. again cluster computing). Hence it cannot be used for a DISPLAYS render grid type of solution.

4.3.3 Loki Render

Loki Render [144] is another Blender render farm implementation applied mainly on Linux based systems; it mostly depends on a TFTPBoot package to function and means all the machines are dedicated to Loki Render set up only (i.e. a kind of computing cluster). The Loki Render requires significant knowledge of computer networking, hardware and software to setup the initial rendering environment as well as requiring good knowledge for any troubleshooting. Similar to solutions presented above, this solution is also good for building a dedicated render farm using dedicated machines, and so this does not meet DISPLAYS rendering requirements (i.e. CPU scavenging).

4.3.4 BURP-BOINC

The BURP (Big Ugly Rendering Project) [145] is based on BOINC (Berkeley Open Infrastructure for Network Computing). It is a wide area network (Internet based) render farm project. The users need to install a small component called BOINC on their system as a service to donate their CPUs cycle to scientific projects. With good research on

various rendering solutions, this is the only grid-based rendering solution found on the web to date. After a successful beta development, the project was sold to NASA (National Aeronautics and Space Administration).

As discussed above, the BURP uses BOINC, which is used for CPU scavenging. The users cannot download a BURP project to build their own render farm, but can be part of a BURP community to use other's CPU cycles. This could have been a good solution if these free cycles were available most of the time, but it has been noted that the waiting time to avail BURP rendering time could be hours or days. Hence, it is not a good alternative to a DISPLAYS render grid, where the developers have full access to render their job at anytime using all available machines.

4.3.5 Deadline

Deadline [146] is commercial software for rendering 3D applications; it allows two rendering machines with a free version of the software. Deadline is a render farm implementation for Windows only applications, and has been developed to be compatible with 3ds Max, Blender, etc. Hence it is a commercial render farm solution, requiring all machines to be dedicated to the Deadline render farm.

4.3.6 ResPower and Rendercore

Yet another alternative: if someone does not want to hold expensive machines, does not require rendering on a regular basis, or does not have enough budget for painless rendering, then one can buy CPU cycles (for rendering their application) from someone else who already has a render farm set up installed. There are many online website who sell CPU cycles for various applications; of course this solution is quite expensive and is not suitable for many low budget organizations.

ResPower and Rendercore are two well known commercial, Internet-based render farms, where the users can buy CPU cycles to render their 3D models and animations. Both ResPower and RenderCore use dedicated clusters and powerful machines for rendering. These kinds of render farms charge on hourly basis, which means if your application is rendered and it takes 20 hours then you have to pay for 20 hours [147]

[148]. Similarly if for any reason you have to render same application again, you need to pay again to render it, hence quite an expensive solution for regular use.

4.3.7 Autodesk Backburner Solution

The backburner solution provided by Autodesk is only for Autodesk applications (i.e. 3ds Max and Maya) that work on Windows, Mac OS X and Linux operating systems. It is a network based rendering environment where many computers connected by a network can work in parallel to render animations (multiple 3D images). The Backburner solution also provides a web-based monitoring environment to check the execution and rendering progress. The Backburner render farm deployment model depends on three components:

1. Backburner Manager: responsible for distributing the jobs on render nodes;
2. Backburner Web Monitor: a client machine that can monitor the job executions;
3. Backburner Server(s): the actual render nodes responsible for rendering 3D animation or graphics.

The Backburner render farm can be installed in either of two modes:

1. Dedicated mode: where multiple render nodes will serve a single application;
2. Shared mode: where render nodes can serve more than one applications.

The Backburner infrastructure concept is similar to a grid infrastructure where all three components (manager, server and monitor) are deployed, although it is usually still referred to as a render farm because you normally set aside a cluster of dedicated machines. Nevertheless, it can be set up to steal idle cycles. It is still not a good solution though, because it is limited to work with 3ds Max and Maya only. Theoretically, the proposed solution is this thesis could be setup to work with many 3D modelling and rendering solutions, and the grid component can be used for other purposes.

4.4 Render Grid System Architecture

The DISPLAYS render grid management tool is a GUI application, designed in Python language, which was chosen for easy integration inside many environments. Another

reason for choosing Python was because it is largely the scripting language of choice for Blender, which means, if required, a toolbar can be created for the integration of render grid inside a Blender 3D environment, although the main objective is to integrate this system inside a DISPLAYS architecture as a part of a DISPLAYS workflow management system.

To prove this render grid concept, seven machines (one manager and six executers) were used to build a test-bed environment (see the implementation Section 4.5 for more details on hardware and software configuration). The render grid architecture includes the development of seven important functionalities, namely:

1. Job Registry
2. Chop Animation
3. Job Submission
4. Render Grid
5. Job Monitor
6. Retrieve Results
7. Encode Results

A complete workflow is designed from submission to retrieval and encoding it to final required format. For more flexibility, the system is automatic; the user needs to submit the job and once the results are ready the user is notified about job completion. The user can also monitor the execution of any job at any time and can interrupt it. Some parts of the workflow are left as manual for the user to decide whether they need them or not. For example, the final output of the rendering process is .JPG images, but the additional functionality includes using encoders to transform these images into an animation or movie format, although the user may want .JPG files in some cases.

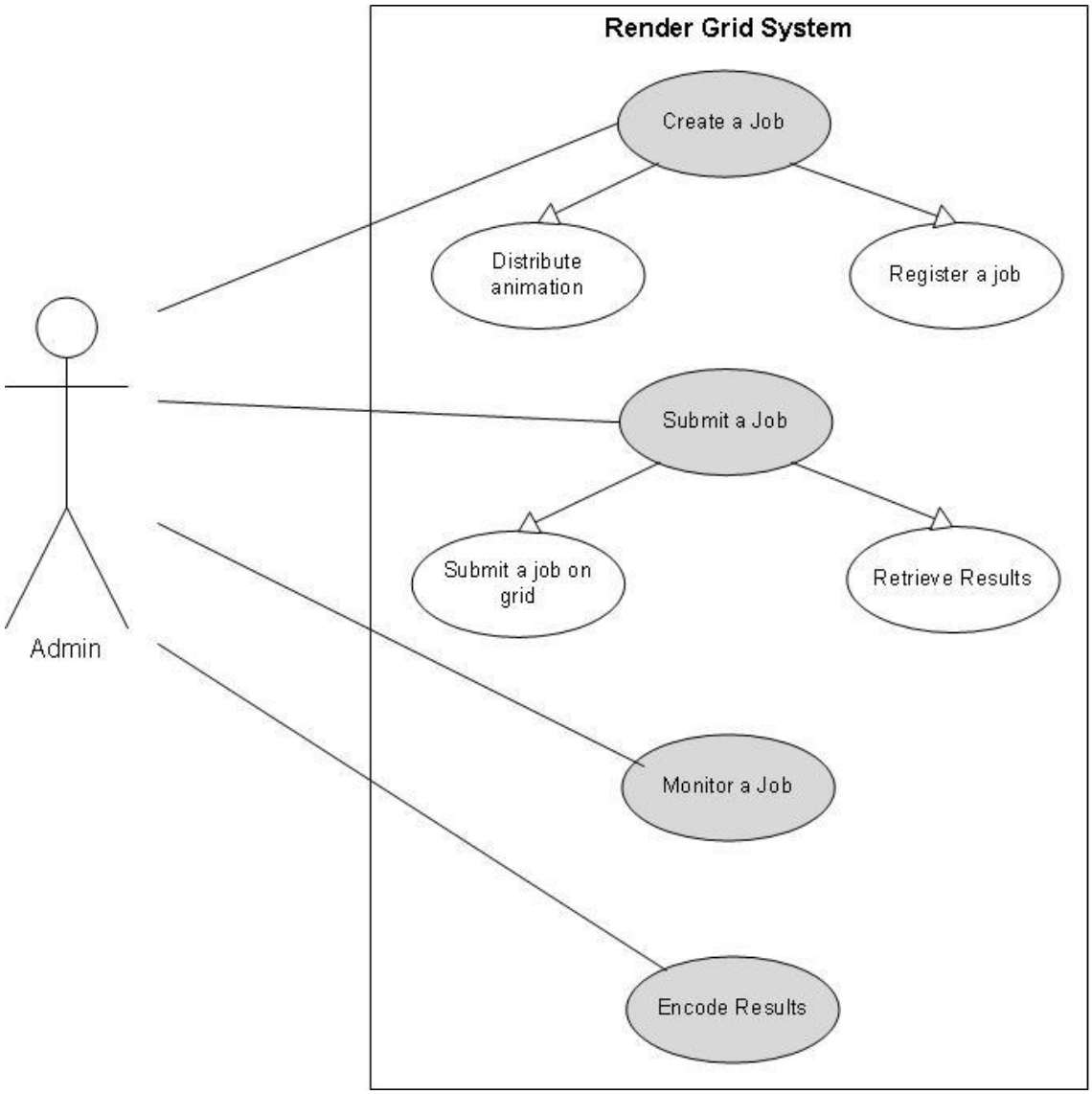

Figure 4.2: Render grid use case diagram

Figure 4.2 presents use case diagram for DISPLAYS Render grid solution, where a user or system administrator creates a rendering job, submits it on a grid environment for rendering. The user can monitor its progress time to time and when the rendering job is finished, the user can then convert the result into required format (e.g. movie or animation).

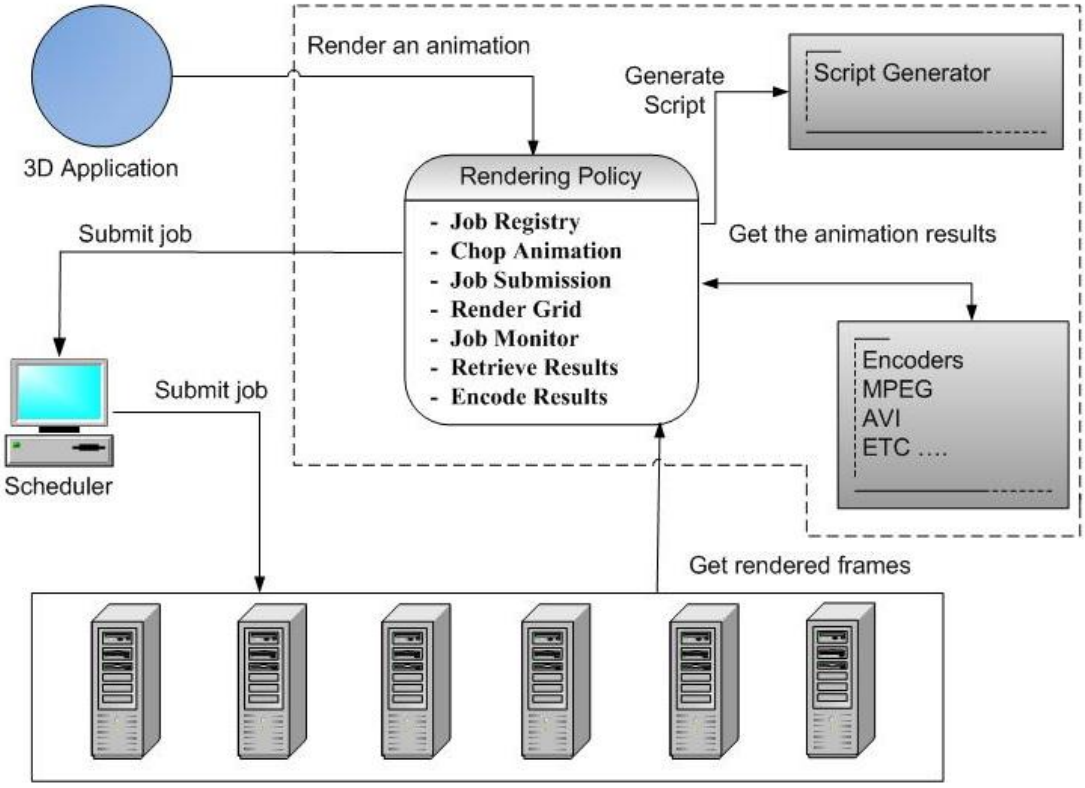

Figure 4.3: DISPLAYS render grid architecture for open source Blender3D

The system architecture presented in Figure 4.3, illustrates the components of the render grid that includes a 3D modelling tool (Blender 3D), a GUI based system management environment (Job submitter, Job Registry, Chop Animation, Render Grid, Job Monitor, Retrieve Results, and Encode Result) and Condor high throughput grid of computers. The diagram illustrates the workflow system designed from the submission of a job using a Blender 3D environment from any submission machine to a central manager (listed as scheduler in diagram above); the central manager or scheduler distributes the job between available grid slaves or servers. The central manager is notified on completion of the job, and the result is returned in a shared folder. The user can retrieve the results or it can further process it using encoders such as FFMPEG [149] to convert into a video format from sequence of images. This system can convert the images from TIFF/ JPG formats to MPG, MP4, FLV, WMV and AVI formats using an open source FFMPEG encoder library.

4.5 Render Grid Implementation Details

The implementation of the render grid involves two main tasks, with many subtasks:

- Setting up the grid environment – so the render grid can be deployed.
    - Computer hardware and network installation
    - Grid software installation
- Implementation of a render grid solution
    - Software implementation of seven components

Below is a detailed description of these tasks in terms of practical implementations.

4.5.1 Grid Computing Environment

The Condor-based grid implementation itself is quite straightforward, the only difficult part is to understand configuration files and manage the machines over the network. There are two conditions when you have to manage grid machines or servers:

1. The servers are non-dedicated (i.e. they are used by someone for other tasks and they are available for the grid when it is free);
2. The machines are dedicated for the grid environment (i.e. machine is only part of a grid as in the case of cluster computing).

In both cases (dedicated or non-dedicated machines), besides the grid computing setup and render grid implementation, the machines maintenance and troubleshooting are also important to keep them running. For example, Microsoft releases regular updates for Windows-based machines, so do Condor and other installed applications; these updates need to be installed from time to time on grid slave and server machines. The grid administrator cannot dedicate a separate monitor for every machine if the machines are dedicated to a Condor pool only and are not used by other users; also the administrator cannot manually apply the updates on every machine.

The alternative is to use any remote administration software such as RealVNC [150], Radmin [151], or Teamviewer [152]. In the case of this research RealVNC was used.

The VNC gives a full control over a remote machine that allows an administrator to remotely manage all machines and apply new configurations whenever required.

The DISPLAYS test grid infrastructure was built with six dedicated computers acting as rendering servers, all containing VNC software for remote management. RealVNC was selected purely because it was found to be a little easier to use than others, although the other two have also been used from time to time for other scenarios (e.g. digital puppetry for digital heritage).

4.5.1.1 *Setting up Grid Environment*

The set up of the render grid solution requires three main components: a Central Manager, a job Submitter and multiple Executers (also called slaves or servers). Just to test the Condor grid middleware it is possible to install all components on single machine, but for an actual render grid solution, more than one executer is required: the same was the case for DISPLAYS render grid. The initial test bed solution contains six machines used to work as executers: one machine as central manager and a personal machine for submitting the jobs in a render grid environment. The functionalities of each grid component can be described as:

1. The ***central manager*** of a grid is a machine that holds the scheduler for the grid that will distribute the render jobs to different executers. The central manager needs to be a powerful machine in all rendering machines. To operate a Condor-based central manager, a user needs to install condor_negotiator, condor_collector and condor_master components on the machine acting as the central manager.
2. The ***job submitter*** is a machine that can submit the rendering jobs to a grid environment. The job submitter machine can be a normal user machine and it does not need to be a powerful machine, except in the case where it also acts as central manager or one of the executers. The job submitter machine can also be used as a part of a grid environment to provide more computational cycles.
3. The ***executer machines*** act as a server, also called a slave, to a central manager that receives the jobs from central manager (grid scheduler). Its job is to execute them and notify the central manager on completion so that the next job can be issued. In the case of Condor, these machines must have condor_start and

condor_master components to provide execution services. Also, these machines must have the application components software installed to execute the required tasks, e.g. the Blender rendering engine for rendering applications.

Below is a detailed discussion on hardware and software specifications and installations.

4.5.1.2 *Hardware Specification and Installation*

As an initial render grid set up to test and evaluate the concept of rendering in a DISPLAYS infrastructure, some available machines were used, as shown in Figure 4.4. All the machines were discarded by the Department of Informatics, University of Sussex due to their expired warranty; the department replaces its machines after a 4-year life cycle. These machines were chosen especially to illustrate how quite old technology can be reused, but also because museums may use just such machines for many years. The question arises: can such old machines have some use left in them? For a test-bed environment seven machines were mainly used, six of them acting as grid slaves (also called grid computing server or executer machines) and one acting as a central manager (also called master). Furthermore, the hardware configuration of these machines was as follows:

- Six grid servers or slaves machines:
  - Four Pentium – IV, 2.80 GHz duo core machines with 512MB RAM, and 40 GB hard drives having Microsoft Windows XP operating system installed.
  - Two Pentium – IV, 2.8GHz single core machines, with 512MB RAM, and 40 GB hard drives having Microsoft Windows XP operating system installed.
- One master/central manager machine:
  - An old Xeon server with 2.40 GHz processor speed, 1GB RAM and Windows XP operating system.

Figure 4.4: The machines used in Render grid setup (one master, six slaves and a router)

Figure 4.4 shows the machines used for an experimental render grid environment: the furthest left is the Dell Zeon server, and the rest are Pentium-IV machines. The other resources include 1 x 32 port router, some UTP cables, and built-in NIC cards. None of the resources are in use any longer, so the prototype has cost us virtually nothing. This experiment also conveys the message of a 'green use' of wasted power from old machines, or recycling the old discarded machines for a good purpose.

4.5.1.3 *Software Specification and Installations*

All the machines had a Windows XP operating system and Condor high throughput software installed and configured as part of the grid. Moreover, for rendering the 3D applications a 3D rendering engine is required; in our case Blender 3D was installed on every machine so that they can work independently with all libraries. In the case where there are some shared libraries or textures, these need to be transferred before the rendering process begins.

In this experiment the complete open source Blender 3D software package was installed on every machine, but one could only install the rendering engine from Blender, as

Blender3D is open source software. Due to this it provides more flexibility to separate the components than other 3D authoring tools.

4.5.1.4 *The Condor Software Installation*

The Condor high throughput solution can easily be downloaded free of cost with its instruction manuals, etc., from the University of Wisconsin website for different operating systems [137]. Condor has a user-friendly installation wizard that takes no more than five minutes to install on a single machine, and then some changes must be made in the 'condor_config' file for proper configuration. It is important that the following changes are applied in the 'condor_config' file after installing it on Windows machines:

1. Select TESTINGMODE instead of UWCS mode set in default.
2. Collector name should be the name of central manager.
3. Hostallow_Read should be set to all names of machines that can view the status of computers' pool.
4. Hostallow_Write should be set to all names of machines that can join the pool and can submit the job to it.
5. Comment out the fields:
    a. NEGOTIATOR_PRE_JOB_RANK
    b. NEGOTIATOR_POST_JOB_RANK
    c. MaxJobRetirementTime.

4.5.2 Render grid Implementation (Core Functionalities)

The actual render grid has been implemented using the Python programming language, which allows easy integration inside many environments. This also allows for integration of this render grid solution inside the Blender 3D environment as a toolbar, so that the open source community can benefit from open source render grid solution. Figure 4.5 presents the GUI interface to a DISPLAYS render grid management, which is currently in progress. The menus are standard Windows environment menus, and at present only 'File', 'View' and 'Tools' menus are active. Additional work is required that includes a tutorial, readme and help files, etc.

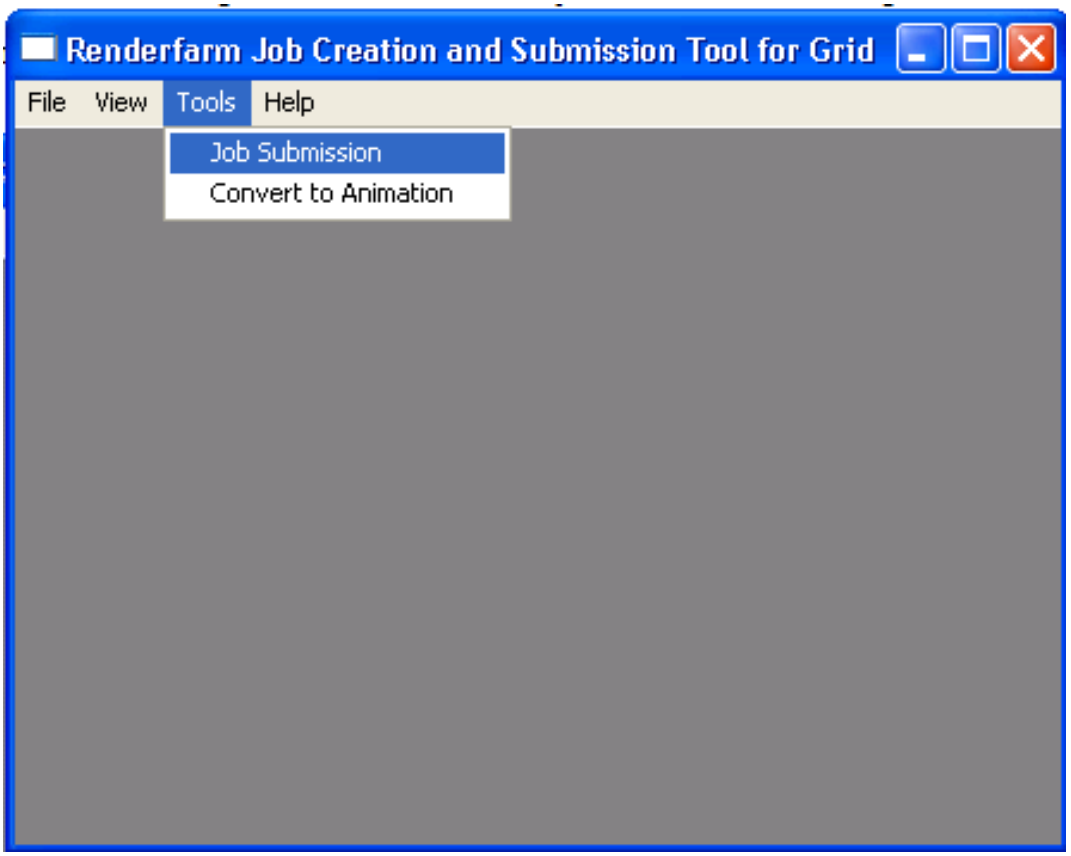

Figure 4.5: Interface to the render grid system management environment

There are seven core functionalities in a DISPLAYS render grid solution as already illustrated above in Figure 4.3. A more detailed description of these functionalities with some important code snippets and GUI snapshots from render grid solution is given below.

4.5.2.1 *Job Registry*

The job registry is an overall database maintained by the render grid management tool where the users can store and manage their jobs to be rendered, and those already rendered by the grid environment for further processing (i.e. format conversion/ encoding to other format and logs management). A lightweight, self-contained, server-less, zero-configuration SQLite 3.0 [153] database has been used (that comes built in inside Python language) for managing user accounts, and their work in progress. The registry only stores the reference to media objects, which resides on a physical hard drive; this keeps the database light and faster to access.

The render grid registry maintains the job information such as:

- Job identification;
- Job name description;
- Number of frames to render;
- Number of sub-jobs from original animation file (i.e. distribution into small number of frames per job);
- Job submission script for grid;
- Job status (whether it has already been rendered or not);
- Rendering results (i.e. path to output files);
- Final output format (if it is compiled to the format, other than default JPG);
- Path to compiled animation.

4.5.2.2 *Chop Animation*

Chop animation chops up the animation into the required number of sub-jobs; it creates a job submission file specifying the number of frames in each job. The chop animation is a component that accepts the Blender animation file with a couple of parameters and generates a grid job submission file. The submission file contains the detailed information about the animation file including name, path to animation, number of frames in an animation, number of jobs and number frames per job to be rendered in the grid environment. The output of chop animation component is a '.sub' file that is stored in a job folder along with the actual animation file and its path is maintained by the registry database. When a job is sent for rendering, this animation file is also submitted to the grid environment along with the submission script. Figure 4.6 is a snapshot of the chop animation component, which accepts the name of the project (it is used for a unique identification inside the job registry for future reference), a complete path to animation file, number of frames to be rendered from animation file, and required number of sub-jobs to be generated for rendering on the grid.

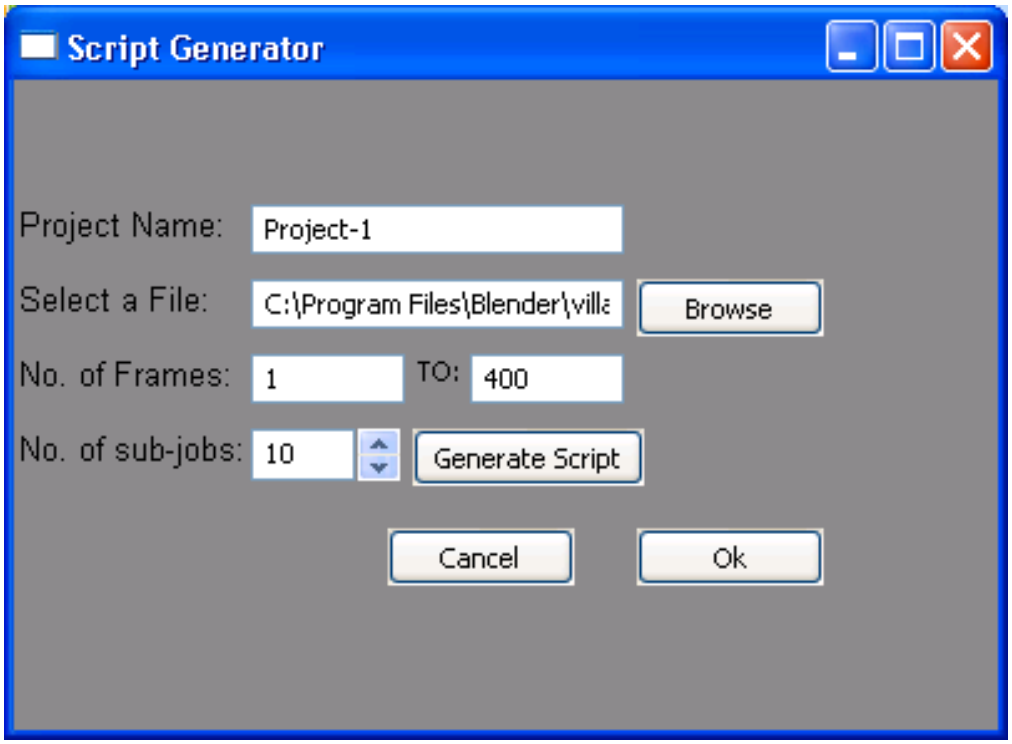

Figure 4.6: Chop Animation also named as Script Generator dialogue box

Although the number of frames inside a job can be computed from a Blender animation file, on many occasions users do not want to render all the frames inside the animation file; this gives flexibility to the 3D artist to decide the frame range to be rendered.

4.5.2.3 *Job Submission:*

Once the job is created by the chop animation method, it is registered in the render grid registry; it is also ready to be submitted on the grid. On job submission, this program submits the generated script along with the original animation file and any other textures or libraries, etc., to the Condor-based grid environment for its execution. This is accomplished by locating the files from the local machine whose path is already available in the registry; these files are submitted to the Condor pool master/central manager for rendering on the grid.

Figure 4.7 is one of several parts of the .sub job file generated by the chop animation component in accordance with the parameters entered (i.e. file path, project name, frame range to be rendered and number of jobs decided from the entered frame range). In the

above example, the user entered 10 jobs from 400 frames, which created 40 frames in each job starting Job-1 from 1-40 and Job-2 from 41-80, and so on.

```
# Render Job 1
Universe = vanilla
Environment = path=C:\Program Files\Blender\
Blender;c:\windows;c:\windows\system32
Should_transfer_files = Yes
Transfer_files = always
When_to_transfer_output = on_exit
Transfer_executable = False
Transfer_input_files = villa_stage5.blend
Executable = C:\Program Files\Blender
Foundation\Blender\blender.exe
Log = log.txt
Output   = out.txt
Error    = err.txt
Arguments= -b villa-5.blend -s 1 -e 40 -a
Queue ()
```

Figure 4.7: Job submission script, for grid environment to render animation file

For testing the render grid environment, we submitted an animation containing four hundred frames, distributed it into ten jobs and rendered it. Please see the results section for more details on performance.

4.5.2.4 *Render Grid:*

The Condor pool master receives a command from the render grid management; the commands are submitted using a command line execution statement from Python with arguments “condor_submit filename” (where the “filename” refers to .sub file generated by the chop animation component). This command is issued when the user selects submit job from the render grid management tool. The Condor master reads the .sub script file, which refers to an animation file and other texture files and libraries. The Condor master transfers all animation source code files to Condor executers together with instructions specified inside the submission script file. It also keeps all the textures and libraries in a share folder, accessible by all executers.

4.5.2.5 *Job monitoring*

The job monitoring component allows the user to monitor the execution of the submitted project in the grid environment. The render grid monitoring tool, see Figure 4.8, allows the users to view and monitor the rendering process on a GUI-based dynamic Gant chart, which works in accordance with executers progress. The job information is gathered using Condor classAd (classified advertisement), part of the Condor installation package, which advertises every system on the central manager for their status. The executers send the signal to the master every ten seconds to update their status. A small package containing status information is passed from the executers to the master, and in turn to the job monitor. Wherever the render grid management tool is installed it displays the graphical representation of the job progress to the end user.

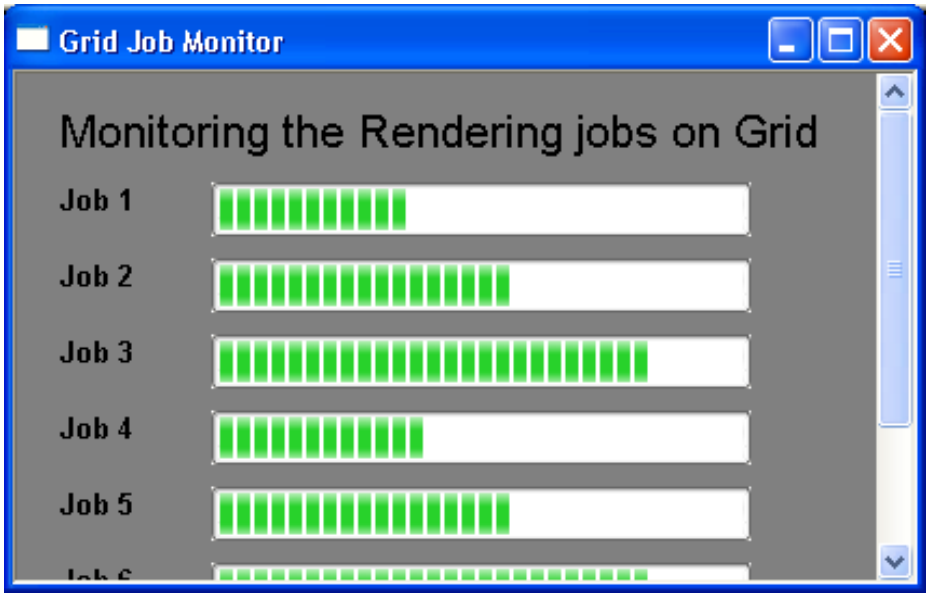

Figure 4.8: Render grid job monitoring tool showing a job in progress on six render machines

4.5.2.6 *Retrieve Rendered Results*

Typically, after the execution of any job on a Condor-based grid environment, the results normally remain on the slave nodes where they are executed unless a method for their transfer is specified within a submission file, e.g: write to shared folder, etc. We have developed our own solution in Python to return the results back to the master for further processing. This is accomplished in a similar way to using a shared folder but it is more robust as it works with monitoring environment. As long as the job is finished

and the render grid monitoring environment is notified, the results are copied into a shared folder on the Condor central manager. For naming convention each image file is named as 1.JPG, 2.JPG, etc., so that the user and system knows the exact frame number. This makes it easy to handle for the next step, which converts these still images into an animation file. Similarly, the name of output folder is exactly same as the project name for consistency with the job description and uniqueness between different job folders inside the same shared directory on the central manager.

4.5.2.7 *Encode Results*

This is an optional but important component inside the DISPLAYS render grid management that allows the rendered output files to be converted into different movie formats. By default all animations are set to generate .JPG files (which are still images/frames) from raw animation, although the end user may want a different format, e.g. AVI or MOV. The 'Encode Results' is an additional step after the rendering process that utilizes an open source FFMPEG encoder to convert a sequence of images into different movie and animation formats. The FFMPEG library can convert the images into most commonly used video formats, such as AVI, MPEG, FLV, MP4, MOV and WMV file formats [149].

The final version of the animation is also maintained by the registry including its path and related information related information (for example, the project itself is marked as rendered to animation = animation file type). This is particularly useful for future references to avoid any duplication in the system.

4.6 Unique Features of this Solution

There has been lot of research in the area of render farm systems for 3D applications; many individuals and research groups have built their own rendering solutions for their projects and some have developed rendering projects for communities. Below are some of the popular render farm solutions found on the web, most of them are open source or freeware solutions; Table 4.1 illustrates their functionalities and differences. The last three are the prototypes of the innovative DISPLAYS render grid implementation presented in this thesis containing all functionalities. At present the DISPLAYS render

grid solution is at the prototype 2 stage, and currently looking for the final version in near future, which will then be available on the web for communities to use.

Table 4.1: Comparison between different available render farms

Render Farms	Features										
	PI	JM	GB	AS	RS	OS	FW	SG	FA	SO	WS
DrQueue	√	√	X	X	X	√	√	√	√	X	X
Farmer Joe	√	√	X	X	X	√	√	√	√	X	X
Loki Render	X	√	X	X	X	√	√	√	√	X	X
Burp-BONIC	√	X	√	X	√	X	√	X	√	X	X
Deadline	X	√	X	√	-	X	X	X	√	X	X
ResPower	√	√	X	√	√	X	X	X	√	X	X
Render-core	√	√	X	√	√	X	X	X	√	X	X
	DISPLAYS Render Grid Management Tool										
Prototype1	X	√	√	√	X	X	X	X	X	X	X
Prototype2	√	√	√	√	√	√	√	√	√	X	X
Prototype3	√	√	√	√	√	√	√	√	√	√*	√*

Table 4.2: Feature Key:

PI = PLATFORM INDEPENDENT	JM = JOB MONITORING
GB = GRID BASED	AS = ANIMATION SUPPORT
RS = REGISTRY SUPPORT	OS= OPEN SOURCE
FW = FREEWARE	SG = SCRIPT GENERATOR
FA = FULLY AUTOMATIC	SO= SERVICE ORIENTED
WS = WORKFLOW SUPPORT	* WAITING FOR CONDOR SOA VERSION

Table 4.1 and Table 4.2 present useful features that a DISPLAYS kind of infrastructure is looking for. At the same time it shows how this solution has a more useful feature list than other projects already available on the web. These features are important as they allow the flexibility of deployment, efficiency, cost effectiveness and an easy to use rendering environment. This kind of open source grid-based rendering solution, which can be deployed on many platforms, is very much cost effective to the end user. An easy to use, fully automated system will hide the actual grid infrastructure from the end user, allowing the user to focus more on application development rather than worrying about more technical aspects of rendering. Below are some important distinctions between other available solutions and the DISPLAYS render grid solution.

- Not all of the render farms listed in Table 4.1 above provide functionalities such as the rendering system being able to be prioritized for foreground users except BURP-BOINC. This provides free services but is currently limited and not fully operational as it does not allow new accounts. In addition, because they provide free services a member has to wait for their turn to get CPU cycles. With our solution, we target an open source grid-based rendering solution rather than a dedicated cluster, which the users can exploit as a rendering environment when their existing computing resources are free.

- Most of the render farms described above, provide either frame-based rendering (individual frames on different systems) or bucket-based rendering (one frame is distributed amongst the systems), or some support both; in both cases the results are generated in image format. Our solution is currently frame-based but the bucket-based rendering program can be developed by adding some existing open source solutions or implementing a bucket-based algorithm into C/C++. The DISPLAYS render grid provides the facilities to get the results in the following formats: AVI, MOV, FLV, MP4, and WMV formats rather than only JPG /TIFF (still image formats).

- The DISPLAYS render grid solution is intended to be built inside Blender 3D as a toolbar for grid and cluster-based rendering environments. With some more

modifications this rendering solution could act as a general purpose portal to a grid environment to submit the jobs and monitor them.

- This solution can also be deployed independently for other 3D applications, such as 3ds Max or Maya, by using web services. This requires working on different rendering engines and executing them though Condor web services, the point here, this is totally possible.

- The render grid's script generator can generate job description files as a job submission for Condor-based rendering. This can also generate execution scripts for other kind of applications that can be used separately on any other grid environments using web services.

- This is an open source and cost effective solution for the scientific and research organizations who cannot afford commercial products due to budgetary constraints for software and cost of new hardware.

- Blender 3D and Condor are also open source, and Python allows binding between these two, allowing the effective integration of a render grid applications.

- The final version of render grid implementation will be service-orientated, and it will be controlled by a workflow management system, allowing flexible and platform independent deployment. This feature cannot be found in any other existing rendering solution.

4.7 Test Results and Discussions

This section presents various tests, results and comparisons between related results to demonstrate the usefulness and effectiveness of render grid solution. These all tests have proved that render grid solution is a useful solution for utilizing idle CPU cycles. Whether it is one machine with 2, 4 or 8 cores or more than one core on more than one machine, the render grid solution has always been effective compared to a non-grid rendering solution. These tests have been carried on small, medium and large size

images on four different type of powerful of machines (Duo Core, Core2Duo, Quad core and Corei7 processors).

4.7.1 Test 1: Grid of Six Computers versus Single Computer with Same Specification

To test the DISPLAYS render grid solution, as an initial test a 3D animation of Fishbourne Roman Palace containing 400 frames was submitted to a render grid and rendered on a grid of six slave computers. Figure 4.9 shows one of the four hundred frames that were rendering in DISPLAYS render grid test environment. This animation does not include very complex illumination, lighting or rendering effects; it was a simple frame-by-frame animation with 400 frames lasting a few seconds. For comparison, it was initially tested on a single machine[3] that took around 7 hours (6 hours and 58 minutes exactly, i.e. 11,665 seconds) to render whole animation and produced an animation file as final output. The same animation was then rendered on a render grid:

- The animation file was chopped up using submission script in 10 equal pieces i.e. 40 frames per job that creates a total of 10 jobs;
- The job was then submitted to the render grid;
- The job was rendered on the grid;
- It took total of 55 minutes (i.e. 3,300 seconds) to finish the whole animation file rendered into JPG images.

This whole process turned out to be 88% faster than submitting the same job on a single machine with one of similar configuration. Later on these images are converted into animation using an FFMPEG encoder included inside the DISPLAYS render grid

[3] This machine had exactly the same configuration as one of the render grid machines.

solution, which took a couple of more seconds to produce a similar animation file as produced by a single machine.

Figure 4.9: One of the four hundred rendered frames from Fishbourne Roman Palace

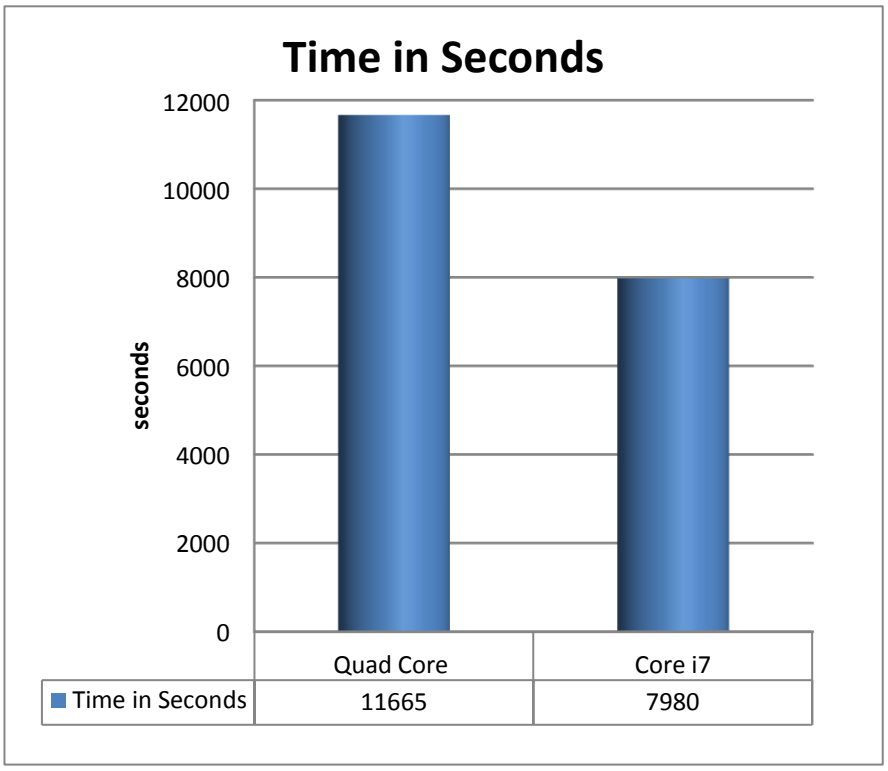

Figure 4.10: Rendering results comparison of grid of seven computers with single computer

In Figure 4.10, the graph shows the performance of a DISPLAYS render grid for seven machines v/s a single computer with a similar specification, which turned out to be 85% faster than single machine rendering. This includes the time calculated for encoding the rendered still images into an animation file. There are some important factors noticed during these experiments; these are highlighted below and one needs to consider these factors while submitting any rendering job in a grid environment.

- Heavy frames to render => less frames per job are preferred, and Light frames to render => more frames per job are preferred:
    - If we have to render an animation containing very high illumination and rendering effects that may take long time even hours for each frame (in some cases) to render then the job distribution should be broken into smaller pieces, i.e. 1-5 frames per job. This is because some of the machines may finish rendering before others and hence they may be idle for a long time while others may still be busy rendering many frames; this is not very efficient in overall performance.
    - Similarly, if the animation file contains very light illumination such as in our case, then a larger size of job would be more effective in terms of performance, as the network file transfer will cause some delay if very small jobs are created.
- Small jobs during working (peak) hours, large jobs during non-working (off peak) hours:
    - The distribution of jobs should be made in accordance with time schedules, as during working hours the grid-based computers are more frequently interrupted by users. If the size of a job is too large, i.e., it may take around 2-3 hours to render, any type of interruption will require the grid to restart the job on another machine; this would also cost more time. Hence the size of job should be smaller in this case (i.e. during working hours for grid-based computers).

- Although during non-working hours, the machines are less likely to be interrupted, so larger job size during non-working hours is preferred, this will reduce network delay, and will improve the overall system performance.

4.7.2 Test 3: Single Quad Core Machine without Grid Computing vs. with Grid Computing Setup

In this setup a single Quad core machine was used to render small, medium and large size images. The idea here was to show how render grid can utilize 100% (i.e. all 4 cores) of Quad core CPU compared none render grid solution which uses 25% (i.e. only one core) of same CPU, hence it executes the same job on same computer but 4 times faster than a non render grid solution.

4.7.2.1 *Small Images on Quad Core Processor*

Small size images were firstly rendered on single Quad processor without any render grid component and it took 37 minutes to render all 400 frames. The same job was then rendered on render grid solution and this time the job was finished in 7 minutes and 10 seconds. This produces an overall improved performance by 4 to 5 times better than the same job on same Quad core processor without render grid solution. . Although a Quad core processor has only 4 physical cores, so in normal case it should only produce maximum 4 times better performance but in this case it is 5 times better results. It was noted that during the rendering process CPU consumption was still more than 50% (i.e. 2 cores in use) when last job was being rendered. This shows that the grid computing environment tries to use more resources in Quad core processor to complete a job more quickly. To confirm this, the researcher performed similar tests experiment with larger size frames, the results are explained in next section. Table 4.6 shows details about the test and Figure 4.14 shows the time difference between two rendering processes.

Table 4.3: Rendering details of small images on Quad core processor

Type of Test:	Quad Core	Quad Core (Grid)
Render Type:	without Grid computing environment	with Grid computing environment
Computer Spec:	Quad core, 4GB memory	Quad core, 4GB memory
Image size:	[200 X 150] pixels, [Width X Height]	[200 X 150] pixels, [Width X Height]
Total number of frames rendered:	400	400
Frames per job	400	50
Number of jobs	1	8
Average CPU usage:	25% – 27%	100%
Start time:	13:38:00	17:18:00
End time:	14:15:00	17:23:05
Total time spent:	37 minutes	7 minutes 10 seconds

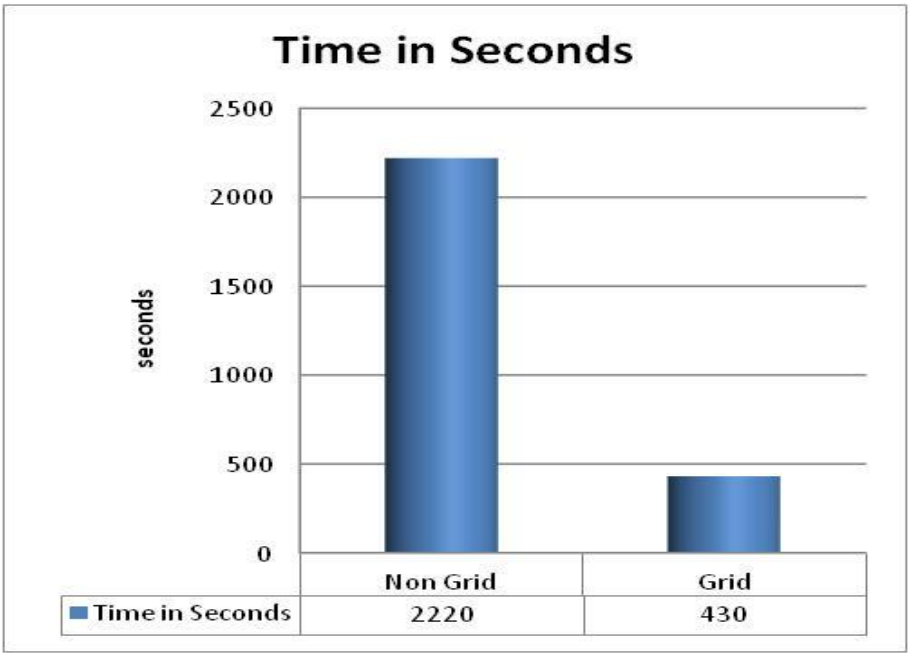

Figure 4.11: Comparison of render grid versus none-render grid solution for small size images on Quad core processor

4.7.2.2 *Medium Size Images on Quad Core Processor*

Similarly, medium size frames were also rendered on Quad core processor first without any render grid component, which took more than an hour to render all 400 frames. The same job was then rendered on render grid solution on the same machine and this time the job was finished in only 13 minutes. This experiment also produced almost 5 times

performance than the same Quad core processor without render grid solution. Table 4.4 shows details about this test with medium size images and Figure 4.12 shows the actual time difference between two rendering processes.

Table 4.4: Rendering details of medium size images on Quad core processor

Type of Test:	Quad Core	Quad Core (Grid)
Render Type:	without Grid computing environment	with Grid computing environment
Computer Spec:	Quad core, 4GB memory	Quad core, 4GB memory
Image size:	[400 X 300] pixels, [Width X Height]	[400 X 300] pixels, [Width X Height]
Total number of frames rendered:	400	400
Frames per job	400	50
Number of jobs	1	8
Average CPU usage:	25% – 27%	100%
Start time:	12:26:30	17:24:30
End time:	13:31:00	17:36:30
Total time spent:	1 hour, 4 minutes, 30 seconds	13 minutes 0 seconds

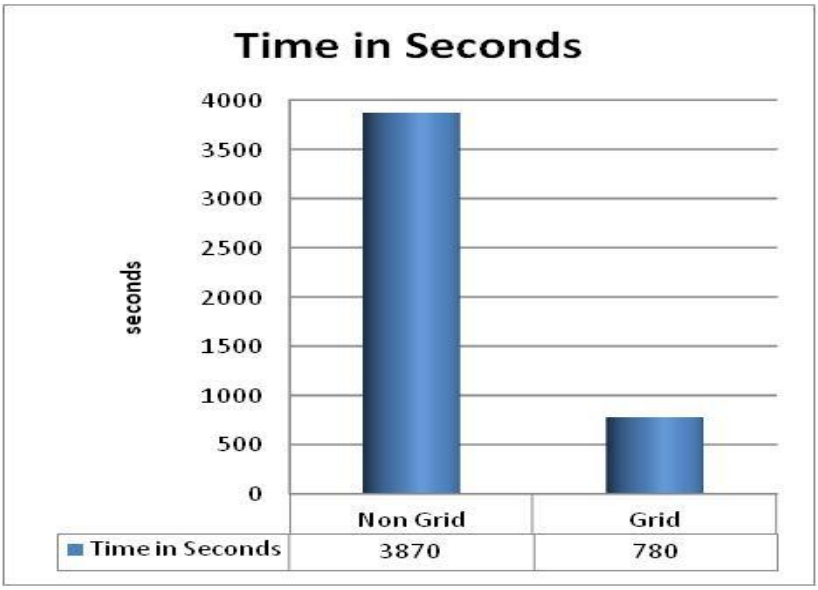

Figure 4.12: Comparison of render grid versus none-render grid solution for medium size images on Quad core processor

4.7.2.3 *Large Images on Quad Core Processor*

To analyze the effect of rendering larger size images on Quad core processor, another experiment was carried out on the same Quad core machine without render grid and

then with render grid solution. The Quad core processor without render grid solution took around 3 hours and 14 minutes. Next day the same animation was rendered on same machine but with render grid solution and this time it took only 38 minutes to render whole animation. Hence again it produced 5 times better performance than non render grid environment. Table 4.5 shows details about this test and Figure 4.13 shows the time difference between two rendering processes. Last two experiments show the render grid can perform 4 to 5 time better performance on a Quad core processor.

Table 4.5: Rendering details of large images on Quad core processor

Type of Test:	Quad Core	Quad Core (Grid)
Render Type:	without Grid computing environment	with Grid computing environment
Computer Spec:	Quad core, 4GB memory	Quad core, 4GB memory
Image size:	[800 X 600] pixels, [Width X Height]	[800 X 600] pixels, [Width X Height]
Total number of frames rendered:	400	400
Frames per job	400	50
Number of jobs	1	8
Average CPU usage:	25% – 27%	100%
Start time:	15:11:00	17:59:25
End time:	18:25:25	18:37:50
Total time spent:	3 hours, 14 minutes 25 seconds	38 minutes 25 seconds

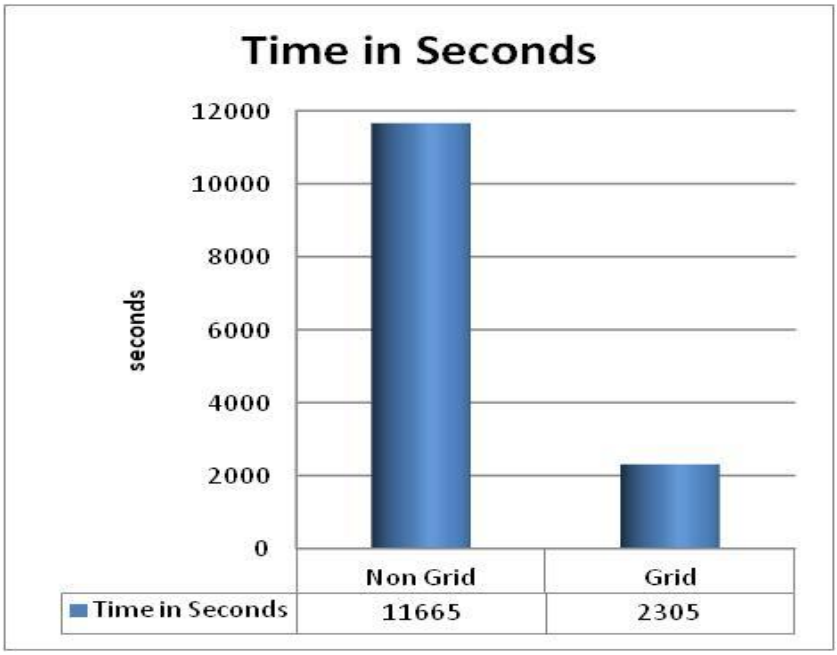

Figure 4.13: Comparison of render grid versus none-render grid solution for large size images on Quad core processor

4.7.3 Test 2: Single Corei7 Machine without Grid Computing vs. with Grid Computing Setup

In this setup a single corei7 machine was used to render small, medium and large size images. The idea here was to show how render grid can utilize 100% i.e. all 8 cores (4 core x 2 threads for each core) of corei7 CPU compared none render grid solution which only uses 12% - 13% (i.e. one core) of corei7 CPU, hence it should executes the same job on same computer but 7 to 8 time faster than in non render grid solution.

4.7.3.1 *Small Images on Corei7 Processor*

Small size images were firstly rendered on a simple corei7 processor without any render grid component and it took more than 27 minutes to render all 400 frames. The same job was then rendered on render grid solution on same machine and this time the job was finished in just 3 minutes and 50 seconds. This produced an overall improved performance by 7 times faster than non render grid solution. Table 4.6 shows details about the test and Figure 4.14 shows the time difference between two rendering processes.

Table 4.6: Rendering details of small images on Corei7 processor

Type of Test:	Core i7	Core i7 (Grid)
Render Type:	without Grid computing environment	with Grid computing environment
Computer Spec:	Core i7, 4GB memory	Core i7, 4GB memory
Image size:	[200 X 150] pixels, [Width X Height]	[200 X 150] pixels, [Width X Height]
Total number of frames rendered:	400	400
Frames per job	400	50
Number of jobs	1	8
Average CPU usage:	12% – 13%	100%
Start time:	15:45	13:11:15
End time:	16:12	13:15:05
Total time spent:	27 minutes	3 minutes, 50 seconds

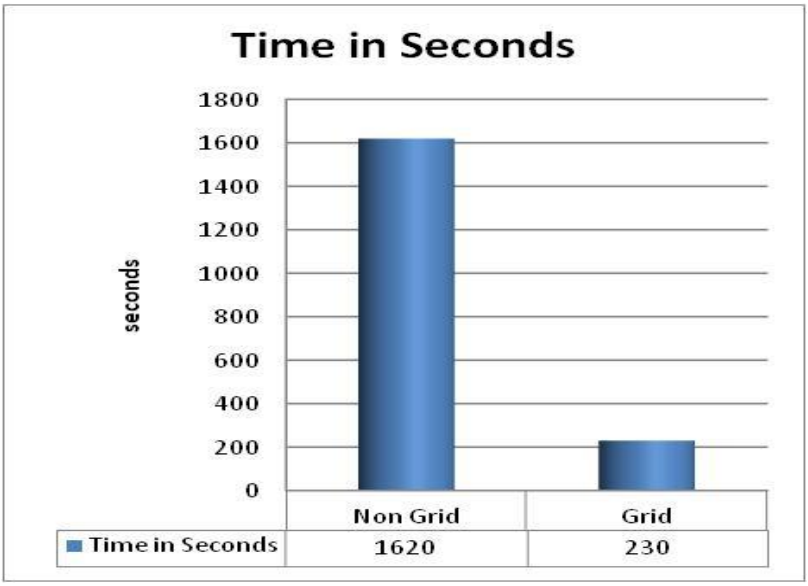

Figure 4.14: Time consumed by a Corei7 CPU with Grid and without Grid computing

4.7.3.2 *Medium Size Images on Corei7 Processor*

Similar to the small size images / animation, medium size images were also rendered on simple corei7 processor first without any render grid component, which took more than 26 minutes to render all 400 frames. The same job was then rendered in render grid environment on same machine and this time the job was finished in 5 minutes and 5 seconds. This again produces an overall improved performance by 5 to 6 times faster than rendering on the same corei7 processor without render grid solution. Table 4.7

shows details about the test and Figure 4.15 shows the time difference between two rendering processes.

Table 4.7: Rendering details of Medium size images on Corei7 processor

Type of Test:	Core i7	Core i7 (Grid)
Render Type:	without Grid computing environment	with Grid computing environment
Computer Spec:	Core i7, 4GB memory	Core i7, 4GB memory
Image size:	[400 X 300] pixels, [Width X Height]	[400 X 300] pixels, [Width X Height]
Total number of frames rendered:	400	400
Frames per job	400	50
Number of jobs	1	8
Average CPU usage:	12% – 13%	100%
Start time:	19:28:10	18:26:20
End time:	19:54:18	18:33:30
Total time spent:	26 minutes 8 seconds	5 minutes, 5 seconds

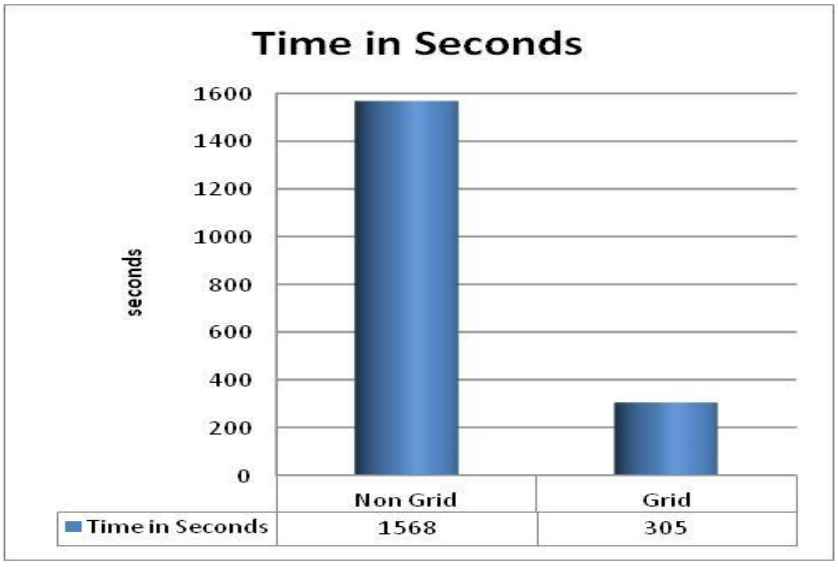

Figure 4.15: Comparison of render grid versus none-render grid solution for medium size images on Corei7 processor

4.7.3.3 *Large Images on Corei7 Processor*

Likewise the large size images were rendered on a Corei7 processor without render grid solution; this took around 2 hours and 13 minutes to complete the rendering task. Next

day the same animation was rendered in render grid environment and this time it took only 22 minutes and 56 seconds to render whole animation. Hence this time it again produced 5 to 6 times better performance than non render grid environment. Table 4.8 shows details about this test and Figure 4.16 shows the time difference between two rendering processes.

Table 4.8: Rendering details of large images on Corei7 processor

Type of Test:	Core i7	Core i7 (Grid)
Render Type:	without Grid computing environment	with Grid computing environment
Computer Spec:	Core i7, 4GB memory	Core i7, 4GB memory
Image size:	[800 X 600] pixels, [Width X Height]	[800 X 600] pixels, [Width X Height]
Total number of frames rendered:	400	400
Frames per job	400	50
Number of jobs	1	8
Average CPU usage:	12% – 13%	100%
Start time:	16:36:00	17:00:00
End time:	18:49:00	17:21:56
Total time spent:	2 hours 13 minutes	21 minutes 56 seconds

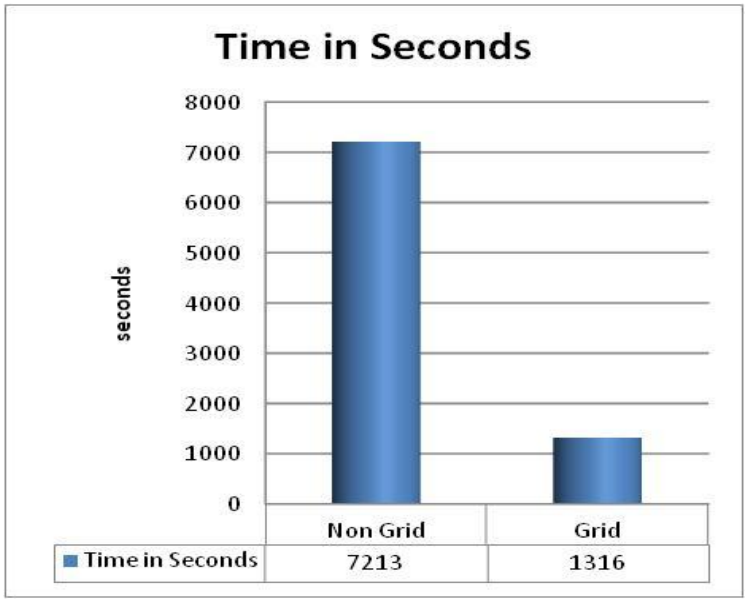

Figure 4.16: Comparison of render grid versus none-render grid solution for large size images on Corei7 processor

4.7.4 Test 4: Render Grid of two Machines (Corei7 and Quad Core) versus Non-Render Grid Environment

Tests 2 and 3 above briefly proved improvement of render grid solution on non-render grid solution; these tests were based on individual Quad core microprocessor and Corei7 microprocessor, where the reason was to utilize all available CPU cycles of single machine. This section is focused on grid of two machines:

1) A Quad core machine (with 4 cores)
2) and a corei7 machine (with 4 cores and 8 threads)

to compare the performance of render grid v/s non render grid solution.

4.7.4.1 *Render Grid versus Non-Render Grid: Small Images*

In tests above small size images were rendered on Quad core and Corei7 microprocessors, where all 400 frames were rendered in 37 minutes on a Quad core machine and in 27 minutes on a Corei7 machine. These same frames were then rendered on a grid of two machines to compare its performance for grid expansion. The rendering was finished in just 3 minutes and 51 seconds. Hence it produced 9 to 10 better performance than a Quad core machine and 6 to 7 times better performance than a corei7 machine without render grid solution. Table 4.9 shows details about the test and Figure 4.17 shows the time difference between two rendering processes.

Table 4.9: Rendering details and comparison of small images on corei7 and Quad core processor with and without render grid solution

Type of Test:	Quad Core (Non-Grid)	Core i7 (Non-Grid)	Grid Environment
Render Type:	Non-Grid	Non-Grid	Grid
Computer Spec:	Quad core, 4GB memory	Core i7, 4GB memory	Core i7, 4GB memory Quad Core, 4GB memory
Image size:	[200 X 150] pixels, [Width X Height]	[200 X 150] pixels, [Width X Height]	[200 X 150] pixels, [Width X Height]
Total number of frames rendered:	400	400	400
Frames per job	400	400	25
Number of jobs	1	1	16
Average CPU usage:	25% – 27%	12% – 13%	2 x 100%
Start time:	13:38:00	15:45	16:12:00
End time:	14:15:00	16:12	16:15:51
Total time spent:	37 minutes 0 seconds	27 minutes	3 minutes 51 seconds

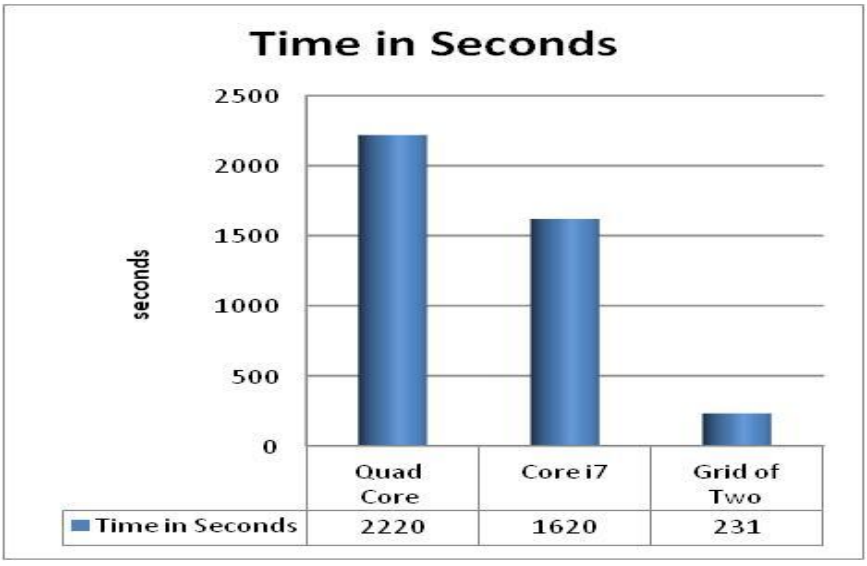

Figure 4.17: Comparison of render grid versus none-render grid solution for small images

4.7.4.2 *Render Grid versus Non-Render Grid: Medium Size Images*

Similarly, medium size frames were rendered on grid of two machines, the rendering was completed in 7 minutes and 32 seconds. The performance comparison was 8 to 9 times faster than a single Quad core rendering and 4 to 5 times better than a corei7

rendering. Table 4.10 shows details about the test and Figure 4.18 shows the time difference between two rendering processes.

Table 4.10: Rendering details and comparison of medium size images on corei7 and Quad core processor with and without render grid solution

Type of Test:	Quad Core (Non-Grid)	Core i7 (Non-Grid)	Grid Environment
Render Type:	Non-Grid	Non-Grid	Grid
Computer Spec:	Quad core, 4GB memory	Core i7, 4GB memory	Core i7, 4GB memory Quad Core, 4GB memory
Image size:	[400 X 300] pixels, [Width X Height]	[400 X 300] pixels, [Width X Height]	[400 X 300] pixels, [Width X Height]
Total number of frames rendered:	400	400	400
Frames per job	400	400	25
Number of jobs	1	1	16
Average CPU usage:	25% – 27%	12% – 13%	2 x 100%
Start time:	12:26:30	19:28:10	16:30:00
End time:	13:31:00	19:54:18	16:37:32
Total time spent:	1 hour, 4 minutes, 30 seconds	26 minutes 8 seconds	7 minutes 32 seconds

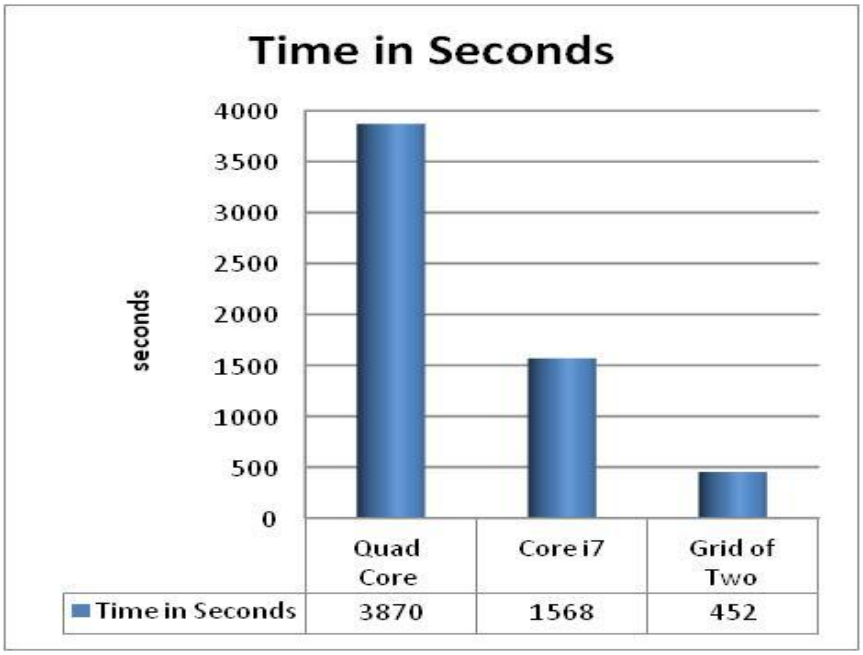

Figure 4.18: Comparison of render grid versus none-render grid solution for medium size images

4.7.4.3 *Render Grid versus Non-Render Grid: Large Images*

Likewise, large size frames were also rendered on grid of two machines, the rendering was completed in 22 minutes, which is again 8 to 9 times better than a single Quad core rendering and 6 times better than a corei7 rendering. Table 4.11 shows details about the test and Figure 4.19 shows the time difference between two rendering processes.

Table 4.11: Rendering details and comparison of large images on corei7 and Quad core processor with and without render grid solution

Type of Test:	Quad Core (Non-Grid)	Core i7 (Non-Grid)	Grid Environment
Render Type:	Non-Grid	Non-Grid	Grid
Computer Spec:	Quad core, 4GB memory	Core i7, 4GB memory	Core i7, 4GB memory Quad Core, 4GB memory
Image size:	[800 X 600] pixels, [Width X Height]	[800 X 600] pixels, [Width X Height]	[800 X 600] pixels, [Width X Height]
Total number of frames rendered:	400	400	400
Frames per job	400	400	25
Number of jobs	1	1	16
Average CPU usage:	25% – 27%	12% – 13%	2 x 100%
Start time:	15:11:00	16:36:00	15:10:00
End time:	18:25:25	18:49:00	15:32:00
Total time spent:	3 hours, 14 minutes 25 seconds	2 hours 13 minutes	22 minutes 0 seconds

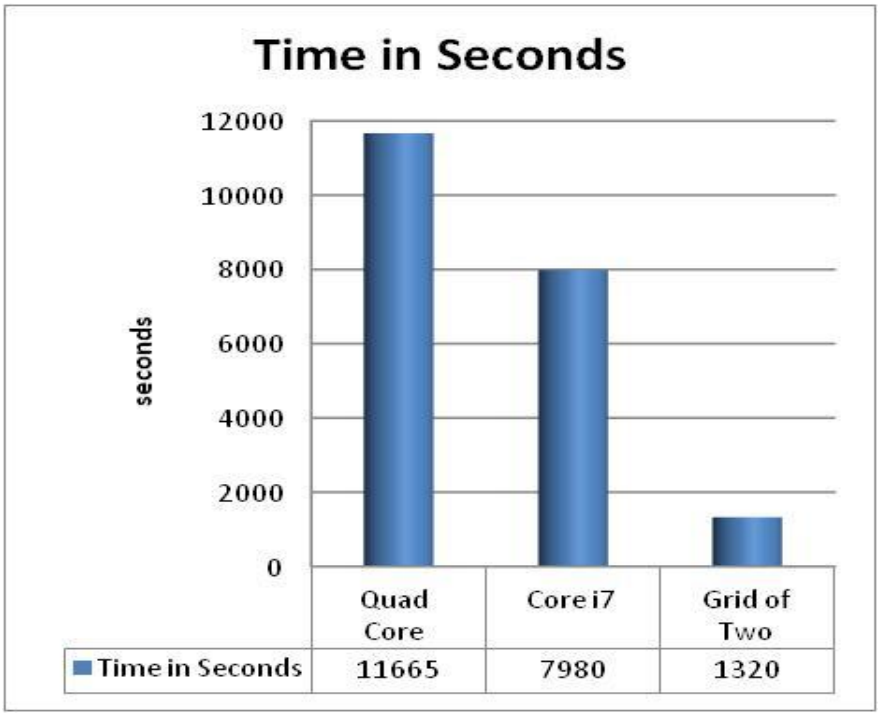

Figure 4.19: Comparison of render grid versus none-render grid solution for large images

4.7.5 Conclusion

Some key points discovered during these rendering were:

- The Render grid solution utilized 100% of CPU power to complete a distributed rendering task, which saves a huge amount of time.
- A Quad core microprocessor produces 4 to 5 times better performance with render grid than a non rendering solution, despite of this that a Quad core has only 4 physical cores.
- Corei7 processor produces 6 to 7 time better performance with render grid solution compare to non render grid solution.
- Rendering larger size images on a corei7 microprocessor reduces the performance in render grid environment but it is still 5 to 6 times better than non render grid solution. The researcher believes, this is because of the memory consumption, as large frames occupy more memory and corei7 renders 8 jobs simultaneously in grid environment. This should improve by increasing the amount memory in corei7 processor.

- While rendering on a grid of two machines some network delays have been observed. This is logical in any network based infrastructure, though it still produces 9 time better performance than a non render grid based Quad core machine. The researcher believes this can also be improved by connecting these machines with a fast network (1 Giga bit per second or more) compared to the test infrastructure of average network speed (100 Mega bits per second).

4.8 Summary

The render grid management tool is a very useful tool or service for the digital heritage library application (DISPLAYS) to provide a community with an efficient DCC service (rendering solution) at low cost. Specifically, the render grid provides the services to those organizations with a low budget, where the institution cannot afford a more expensive computing environment (e.g. a render farm of cluster or blade with associated commercial software packages). Also, if low budget organizations build their own grid environment for rendering, they can also use it for other applications to fulfil HPC requirements. Hence this will reduce the overall set up and running costs. All the tools used in this project are open source and freeware, and the adoption of this system should be virtually free of cost, except some old or already in use machine power.

The DISPLAYS render grid management tool includes a fully automatic process from building a job to retrieving the results in the required animation format. The management tool hides the grid and other functionalities such as Job Registry, Chop Animation and Job Submission; it provides more control and management to the overall system.

The test results show that this kind of set up is efficient in performance, especially only using seven old computers, prove that a render grid would be useful for 3D modellers to utilize all machines already available in an organization to act as a render farm. This innovative result, the DISPLAYS render grid, is discussed further in peer reviewed work [114] [113].

Moreover the additional test results on Quad core and corei7 processor has proved that render grid is not only useful to network based machine but it can also be used on single machine having more than one core to achieve 100% CPU utilization.

Finally, this chapter demonstrated an example of one of the DISPLAYS type components that find use in a digital heritage system, the following chapter describe another component which allows the users of digital heritage systems to interact with digitized cultural heritage environments, such as: the one rendered in this chapter. These both research works presented in chapter 4 and 5 are part of DISPLAYS framework, as described in chapter 3.

CHAPTER V

5 Interaction through Motion Capture

5.1 Introduction

The aim of virtual exhibition for cultural heritage is to provide the user with the experience of life in ancient times by visualizing the virtual sites, monuments, objects and other life presenting characteristics such as audio, video and text elements. Most of the projects in digital heritage relate to virtual tours and heritage site reconstruction that are either online systems (such as a website or portal) or studio systems (such as cave systems) designed to visualize only some objects (such as a piece of clay, stone or any other portable antiquity) or a heritage site (such as a church building) with a very small amount to view to demonstrate the presence of life and with very little or no interaction between user and environment [154].

Generally cultural heritage resides in three places:

1. the actual objects in museums;
2. their stories in books; and
3. their location, which may or may not even exist anymore.

Where heritage objects are located inside a museum they attract many visitors, but they do not retain their attention for long, i.e. they do not look as attractive as they should as they look lifeless, do not present their story and, in fact, they often cannot be observed (when in conservation or storage) or touched (when on display) due to their fragility. Sometimes books (that can be purchased in the museum gift shop) present their story,

but more often the objects museums label are the only public accessible knowledge available. This does not provide the user (museum visitor) with a very comprehensive vision of the object that could be provided given the modern technologies that could be used. The full story is not shown, where is the historical context behind the object? Furthermore, the location where the object originated, what is this story? How did people interact with the object and location? The full story is far greater that the museum label!

Think of a character from an ancient world being inside his own world showing and explaining his own life, his behaviour and the customs of his times. He is talking to the people and the people are asking him various questions; he replies to them all in real-time. Think of a broken antiquity inside a museum being simulated in its original (virtual) space, and here this object is not broken and is being shown in use by its owner who is explaining its importance in his life to the museum visitor. This is all possible using an old technique called 'Puppetry' by combining it with state-of-the-art technologies such as motion capture systems, 3D graphics, audio, video, text and some programming techniques for lips, facial and body movements. Hence, the new technique is called 'digital puppetry'. Here, the unique innovative feature presented in this thesis are the integration of real-time online motion capture technology with digital puppetry that is now achievable through the use of eMove motion capture suits and associate software (e.g. real-time gaming drivers, etc.).

The objective of this research in the area of digital puppetry is to implement a real-time online digital puppetry solution for digital heritage, where different possible scenarios can be achieved to make static and scattered heritage into a dynamic and alive heritage world. This concept also leads to the concept of virtual time travelling to an ancient world for interacting with ancient humans, their environment and their belongings.

The use of motion capture suits with 'joysticks' inside games engines provides the possibility for a museum visitor to walk through the inside of a heritage world and interact with heritage objects. A user simply puts on an eMove motion capture suit and connects to a game engine that has the heritage world replacing a normal game

environment, and then navigates and interacts with the heritage environment in a similar way to a game.

5.1.1 Digital Puppetry

Digital puppetry is a kind of real-time computer generated animation (different from passive or sequential animation). It is an art performance, played and derived by a real actor who understands the art of acting the role on a digital stage.

Some of the terms associated with "digital puppetry" are defined as:

- ***Digital Puppetry or Digital Puppetry*** is computer generated real-time animation played and controlled by an actor in the real world;
- ***Digital Puppeteer or Puppet Master*** is a person who drives the digital puppets;
- A ***Digital Puppet*** maker designs 3D or 2D models of a digital puppet.

Digital puppetry is different from simple computer animation, where the user has no control during the execution of the animation, except for some predefined actions such as pause, play, rewind, forward and stop. However, digital puppetry needs to be derived by the human controller at run time either using joysticks, keyboards or, in the case of this thesis, through the integration of real-time motion capture technologies with appropriate gaming engines. Here, when the human moves, its corresponding puppet also moves. Advance versions include a mix of all the above technologies i.e. motion capture system with additional joystick or keyboard keys to trigger motion blending for different facial expressions such as eye wink, smile, happy and sad facial expressions, etc.

Digital puppets are digital performers who act as real actors to entertain their audience; they are lifeless without a proper voice, facial emotions and body movements. An online real-time digital puppetry application requires many additional research areas to work with (such as animation, motion blending, lip-syncing, voice and 3D graphics) in order to create an effective digital puppet. This research is focused on implementation of an innovative and robust online digital puppetry solution for digital heritage that can

fulfil the requirements of the DISPLAYS framework. Several basic key elements are needed to bring about an online digital puppet that can implement a real-time heritage character as outlined in Chapter 1: Interaction through Motion Capture', and described in Chapter 3:

1. The ability to map motion captured data onto a skeleton structure or rig (bones) that can then be animated in real-time;

2. The skeleton has to be able to be characterised with a character (heritage character in this case);

3. The heritage character needs to have a so-called 'talking head' that can be animated in real-time through facial animations;

4. The talking head needs to exhibit lip-syncing in real-time;

5. The system has to have networking support to enable remote connections between the motion capture suit and operator and the audience;

6. The system has to present the heritage character in a virtual environment that depicts the age.

Some obvious technologies present themselves for implementing a solution to this scenario, 'Interaction through motion capture' with a heritage character. A typical gaming engine satisfies requirements 1 and 2, the integration of MotionBuilder also satisfies 1 to 4, but no complete solution exists to present these solutions online. This chapter proposes a system that provides a first prototype solution that proves the concepts outlined in this scenario, and which utilizes components such as MotionBuilder integrated with the eMove SDK and bespoke software, e.g. eMove-chat. This chapter describes the implementation details of this prototype and its evaluation, thus paving the way for future more efficient implementations currently being considered; see future work in Chapter 6. Additionally, some part of this research suggests possibilities for reducing its cost and improving its performance. Also, some other possibilities from this work are discussed such as walking through a digital

heritage environment and playing with digital heritage objects in virtual environments; explained through scenarios.

5.1.2 Related Work

Although the scenarios presented in this research work are quite new to the digital heritage world, other related work, such as ERATO and CHARACTERISE, should be mentioned because it sets the scene for the scenario implemented in this chapter.

5.1.2.1 *ERATO*

Work such as that done by the ERATO (Identification Evaluation and Revival of the Acoustical heritage of ancient Theatres and Odea) project [155] [156], illustrates the concept of including a pre-recorded character animation within a virtual heritage scene. However, pre-recorded character animation does not allow for dynamic indicative dialogue with a visitor, nor does it allow a visitor to 'enter' the heritage world through 'embodied interaction'.

5.1.2.2 *CHARACTERISE*

The CHARACTERISE (Virtual Human Open Simulation Framework for Cultural Heritage) project [157] [158] focused on a toolkit around already available technologies that can allow heritage world developers and modellers to place intelligent and multilingual 3D avatars into a virtual scene. These avatars are modelled using some avatar creation software and speech synthesis (artificial or computer generated production of human voice usually through text-to-speech TTS software) will be added to these avatars. In the same way as the ERATO project, it does not allow for dynamic indicative dialogue with a visitor, nor does it allow a visitor to 'enter' the heritage world

5.1.3 Limitations of Existing Heritage Projects

The research covered in this chapter for interaction through motion capture differs remarkably from both the ERATO and CHARCTERISE projects because this research replaces pre-recorded animation and dialogue with real-time motion capture based animations and digital puppetry, thus allowing an expert narrator and actor to assume

the persona of the character of the age. The implementation of this heritage scenario is possible due to the innovative development of the eMove – Personal Motion Sensing System project [121].

Using eMove technology and its SDK, it is a short step to the creation of a reconstructed heritage world and placing a real-time avatar (controlled by a museum officer or a real actor) into that heritage world. Furthermore, using eMove motion capture suits, visitors can immerse themselves into the heritage world and interact (navigate, touch, pick up and move objects) with their digital heritage equipment using the joysticks on the eMove suit. This involves almost everything in real-time including motion capture along with audio, lip movements and facial expressions.

To summarise, existing digital heritage projects have several limitations in their interaction and presentation parts:

- Some of the existing digital heritage systems provide interaction methods, but these methods are limited to only passive fly through, which is kind of virtual tour with play, pause, rewind and forward option; they do not provide any dynamic interaction mechanism such as proposed in this thesis.

- Only one person can experience the heritage virtual tour or passive fly through in a virtual environment, having more than one end-user is not supported in existing digital heritage systems.

- In general, only traditional interaction tools are used such as joysticks, keyboard, etc. They do not integrate and interface with other virtual reality technologies (such as human motion capture, games engines, etc.).

- Although EPOCH [159] [160] has introduced some real-time interaction methods, these methods are only limited to rotations of a virtual object using its physical replica. There is no human gesture control or dynamic walk-through inside the heritage environments; neither can this be used for a digital puppetry type of application. The same interaction can be performed with a combination of motion capture suit and joysticks and with the help of an innovative grip

system, which has been already tested inside Unreal game engine (discussion on virtual grip system is outside of the scope of this thesis).

- Most heritage systems do not provide the experience of role-playing inside a heritage environment, which gives a visitor to the heritage interaction the ability to assume a different person in the form of a role-based avatar inside the heritage environment.

- Usually, only one person experiences the interaction with a heritage environment, whereas with the scenario proposed in this thesis group interactions are possible.

5.2 Motion capture for Digital Puppetry

Digital puppetry has an old relationship with motion capture technologies: there have been many efforts to build a digital puppetry application using motion capture technologies for animation and performance. Digital puppetry systems, where motion capture systems are used, is a 2D or 3D avatar, a virtual representation of a real human; behind the scene a real human wearing a motion capture suit (or sensors) controls and drives a virtual character. The motion capture suit contains various sensors placed on different parts of the human body, which detect every movement of the human. These movements are transformed into computer signals that trigger various animations rendered in real-time on a computer screen.

Motion capture systems are widely used in virtual reality applications; in fact the most common applications of motion capture systems are virtual reality character animations such as digital puppetry, cartoons and the film industry. In most cases the motion capture data from motion capture systems is recorded using a standard such as BVH (Biovision Hierarchy) and is fed to a digital avatar. This is far better and more realistic than a traditional joystick that can trigger some predefined animations. Using a motion capture suit to drive a digital avatar gives more control to the person wearing the mocap suit, as he can dynamically generate many realistic animations in real time; it may take hours, days or even weeks to generate similar animations using any other alternative methods such as frame-by-frame animation, etc.

Historically, the concept of digital puppetry using motion capture system is not new; experiments have been conducted for many years by many organizations and groups. Some popular examples that made the changes in motion capture for digital character animations are given below.

In 1962, Lee Harrison III tested digital puppetry using sensors technologies using an ANIMAC hardware contraption; Lee Harrison also developed the ANIMAC in 1960. This attempt is generally recognised as the first attempt at digital puppetry applications [161]. Lee Harrison and his group animated a stick man character on a CRT screen in real-time; the stickman was connected with motion tracking sensors placed on human body.

Figure 5.1: Lee Harrison III, photo montage featuring a dancer with body mounted sensors controlling real-time animation on the ANIMAC, 1962, Denver [161]

Lee Harrison is known as a computer graphics pioneer, and he was also awarded the National Academy of Television Arts and Science Award for outstanding achievements in engineering and development in 1972.

In 1985, Jim Henson Productions from Pacific Data Images (PDI) tried to develop the computer graphics version of their character (a digital puppet) for real-time animation, but they had limited success in their work due to technological limitations. However, in 1988, the availability of silicon graphics 4D chips allowed the Pacific Data Image expertise to produce a decent solution using custom eight degree of freedom input devices. The Jim Henson production was able to control the position and mouth movements of a digital avatar. One year later in 1989, Pacific Data Images produced the first upper body mechanical (exoskeleton) motion tracking suit. The lightweight motion tracking exoskeleton was capable of tracking head, arms and torso in a human body [162].

Obviously there are other methods available to drive a digital avatar, such as a keyboard, mouse, joystick, etc. as discussed above, but they all involve generating predefined animations (key frame animation) controlled by keys. These methods are fine for cartoon characters but they are quite cumbersome if used for real life characters where many dynamic movements are needed.

It is quite clear that motion capture is the immediate and quicker method of choice by many productions for implementing digital puppetry applications for virtual representation of real life characters, and so it has been selected for this research to build a digital puppetry application for a digital heritage framework.

5.2.1 eMove – Personal Motion Sensing System

The eMove – Personal Motion Sensing System is a motion capture project involving various tasks including providing a low cost state-of-the-art motion capture system at the consumer level. Most of the motion capture systems to date are extremely expensive and not everyone can afford them, these systems are highly technical to operate and only professionals can afford these mocap systems for building motion contents for game development, TV or film production.

The eMove (Personal Motion Sensing System) is a collaborative project between Animazoo UK [86] and the Computer Graphics Centre at the University of Sussex [163], and is funded by the UK's Technology Strategy Board [164], creative industries digital technology theme and Animazoo. A major goal of this project is to produce personal motion capture system for consumer level applications for about $300 rather than thousands of dollars.

Figure 5.2 below shows, on the left hand side, an initial illustrated eMove suit and on the right hand side the CAD design for an actual eMove suit expected to be released around the middle of 2011.

Figure 5.2: eMove Personal Motion Capture Suit: on left initial illustrated version of the suit, on right side CAD design of final version of eMove suit.

As a part of the eMove project, three scenarios are being developed with various applications including online real-time digital puppetry. The eMove scenario of concern to the work in this thesis is focused on building a digital puppetry system for general entertainment and theme park applications. The innovation proposed here is the adaptation of this work towards a novel heritage-based 'interaction through motion capture' system. This application integrates the eMove motion capture system, a new voice over IP application; high resolution 3D models, character animations and animation software, and facial expressions with lip syncing that can be used to build any digital puppet application, not just a heritage character scenario.

5.3 Interaction through Motion Capture Heritage Scenario

Cultural heritage objects, their surrounding environment (where they actually lived), and the stories about their usage from books, presents an essence of life from ancient worlds. This section presents three scenarios where motion capture technologies using digital puppetry types of application can be of benefit to cultural, digital and museum heritage communities.

5.3.1 Heritage Role Play Show

This scenario is about redrawing (modelling) cultural heritage objects, putting them in their original environment, sketching their owners (users), building a heritage world, and reanimating cultural heritage through the knowledge retrieved from books to present the life, behaviour, objects, and their usage in ancient period.

Scenario: A museum has their objects archived in physical form but they are not complete: many objects or parts of objects are missing, their users are missing and, in fact, the whole ancient environment where these objects were actually present is missing from the museum's collection. A heritage research group organizes a heritage play, a kind of theatre show with virtual assets and environments. Some 3D modellers are arranged to reconstruct the whole virtual environment (e.g. Fishbourne Roman Palace) where these all objects are placed; the object and human models are also designed. After modelling all objects, human models and their surrounding environments, these are rendered using DISPLAYS render grid functionality. The whole rendered environment containing all assets are imported into a game engine (such as Panda3D, Unity3D, Unreal, or an animation tool such as MotionBuilder). The 3D virtual avatars are connected to the eMove motion capture suits using the real-time eMove mocap drivers (part of this work). The director of the Fishbourne Roman Palace heritage play arranges many local theatre actors to act as various virtual characters inside the heritage play. The actors wearing low cost motion capture suits play various parts (one being an Emperor and so on) inside the virtual Fishbourne Roman Palace (digital environment). These characters can speak with each other having proper lip-syncing and facial expression, they can walk inside the heritage environment using

joystick keys on the mocap suits. They are able to hold the virtual object and perform a character's routine tasks insides the virtual environment as they would in real life. The whole play is recorded with sound and all animation steps and at the end of the play, when the audience has returned to their homes, the heritage play is compiled into a video format to present on either the museum's kiosks, archived on the museum's website, or played as a background video for digital puppetry (explained in the next scenario).

5.3.2 Digital Puppetry

A typical DISPLAYS museum interaction scenario starts when a group of museum visitors (e.g. an organized educational school trip, some archaeology fellows or a family) visit a museum or ancient archaeological site to view the museum's physical cultural heritage and experience a new virtual heritage world. Having wandered around the museum, the group can actually see many heritage objects; they now want to see these objects in their original place with their role in the heritage world. The group then comes to an immersive heritage interactive gallery that promises to take everyone on a virtual tour of ancient history where they can meet with people in the ancient world and interact with them.

A group of visitors entering into a virtual world meets a virtual 3D avatar from Roman history on either 3D or 2D high definition or a simple LCD screen; this could be a bishop of an ancient church, Emperor of some ancient world, a Queen or one of her slaves. This is a place where ancient characters can actually talk to the group of people about their ancient world in real-time. They are amazed to see a heritage character such as 'an Emperor of Rome' appearing in front of them on a display screen who is engaged in dialogue with them, his lips are synched to his voice, his eyes are blinking as a real human, and his facial expressions on different occasions present him as an alive character. The Emperor seems to be able to see who they are (his audience), he introduces himself to the visitors and asks them many questions such as their names, where are they from, etc., all of this happens in a natural way as in real life and not through a pre-recorded animation. The avatar's sound is modified through the pitch controlling software to match with the character. The Emperor can then show them

some virtual objects (perhaps some diamonds or a necklace from his treasury) and describes these objects and their characteristics from history. Figure 5.3 below, on the left hand side demonstrates this scenario, where a group of people is interacting with a digital heritage avatar.

The Emperor takes them on a virtual tour of his palace with people talking and interacting with each other (this could be a pre recorded animation from the previous scenario), it may appear behind the Emperor (a background video) on a digital screen; this video could be about life in the palace. The Emperor explains to them about their lifestyle and introduces their culture, behaviour, customs and food. The engaged group of visitors may ask many questions of the Emperor: a visitor from the group may ask about his crown and the Emperor replies to the question dynamically, turning his attention to the visitor asking the question thus developing a rapport. This whole scenario can be a learning activity about a particular heritage environment, its objects and people living inside that environment.

The real-time character animation is operated by a person wearing the eMove inertial motion capture suit in the next room, behind the screen or curtain, or from a remote location. Although the operator can be remote, he sees the visitors through a web camera and speaks to them through a microphone and speakers mounted near to the display screen. The eMove chat application (part of this research) has been implemented specifically for digital puppetry applications such as this application.

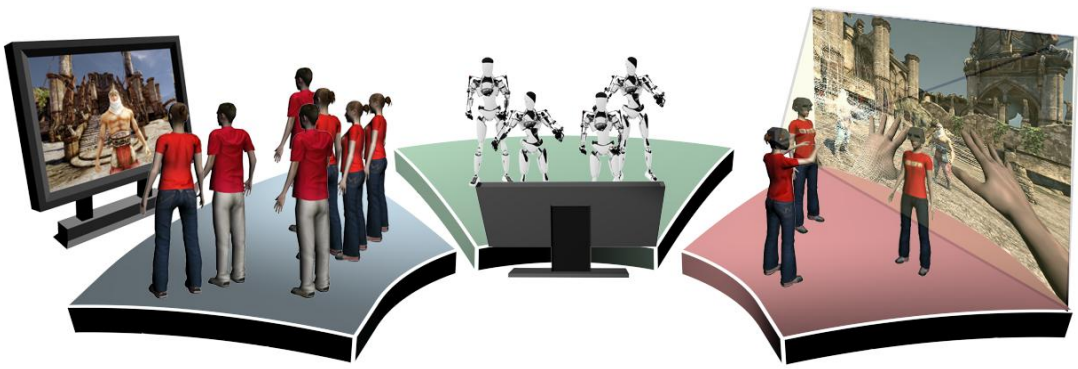

Figure 5.3: DISPLAYS Digital Puppetry and Real-time Heritage Interaction

5.3.3 Audience Interacting with the Virtual World

This particular scenario can be extended by adopting the first scenario in Section 1.5.1; however, instead of actors wearing the eMove suit, the Roman Emperor asks the visitors to join him in his heritage world. The museum visitors then move over the eMove suits shown in the centre of Figure 5.3 and put them on. After the start up process, the visitors shown on the right hand side of Figure 5.3 now appear in the ancient world. The character who was engaged with them at the beginning will continue to guide them. Now the visitors can see one another, what's more they can reach out and touch each other, pick up objects, and they can interact with the heritage world virtually. The group can now start to walk around the heritage world using the joysticks on the motion capture suits for navigation controls, and, using the systems headphones, they can hear the sounds from the virtual heritage world.

5.4 Specification and Design of the Digital Puppet System

The digital puppetry application for digital heritage uses various state-of-the-art technologies outlined below. The design phase for digital puppetry follows the object oriented analysis (OOA) and object oriented design (OOD) techniques.

The design phase covers task execution and various use case scenarios; on the basis of these descriptions the actual application has been implemented. Also, the overall system architecture for this work is explained in this section, which includes all hardware and software components and client-server architecture for this work. During the rest of this chapter, the generic 'Interaction through motion capture' scenario from which the specific scenarios discussed above is implemented through the 'digital puppet application'. Thus, the remainder of this chapter now discusses the implementation of the 'digital puppet application'.

5.4.1 Digital Puppet Task Execution Sequence

The digital puppet tasks execution can better be defined by UML's sequence diagram, which is kind of an interaction diagram that explains how the processes are operated, what their sequences are, who their initiator is and where they terminate.

The digital puppetry application has been designed using different hardware and software components. The hardware components include the eMove motion capture suit, computer system(s), and their connectivity. The software components listed in Figure 5.4 are part of the eMove SDK (the Animazoo SDK modified as part of this research work). The sequence diagram below describes the tasks initiation and execution sequences for this digital puppetry application.

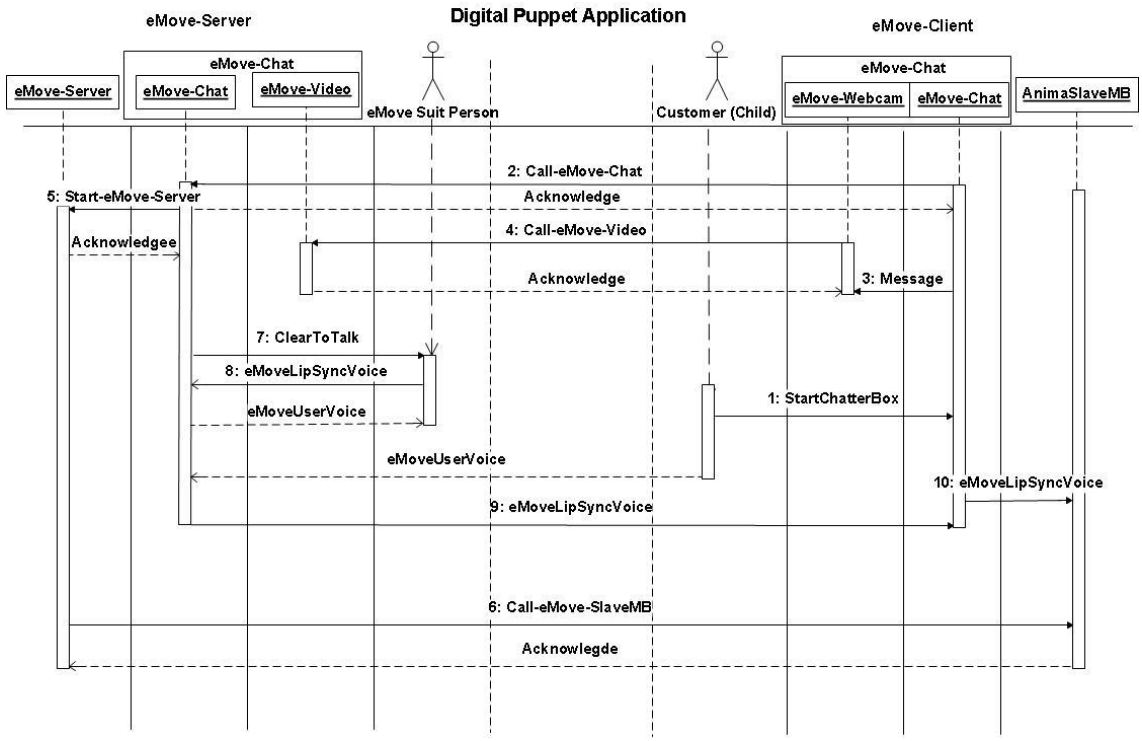

Figure 5.4 Digital Puppetry Sequence Diagram to Initiate Client and Server Communication

Broadly there are three main tasks that need to be initiated by the users, and other sub-tasks are automated by the system. The user tasks are:

Start voice call	⇔	Accept the call
Client: start webcam	⇔	Server: start video
Server: Send mocap data	⇔	Client: accept mocap data

Other sub-tasks are initiated internally as a system process; the end users do not need to worry about these sub-tasks.

5.4.2 Use cases

A use case outlining the digital puppet operation is defined; see Figure 5.5, which Figure 5.5 shows the DISPLAYS use case diagram for the digital puppet application. There are many things that can be covered in this description but mainly it describes the originator of a process (who can do what), a process description, pre-conditions, post-conditions (output). There are nine use case scenarios designed for this research and these are listed below. The detailed description of each of these scenarios is quite lengthy, hence these use cases can be found in Appendix A of this thesis.

1. Start the 'online animation, audio and video chat system' (client side)
2. Start the 'online animation, audio and video chat system' (server side)
3. Select 3D Avatar
4. Call to 3D Avatar
5. Audio, Video, Text Chat
6. Change 3D Avatar
7. Joy Stick Actions
8. Changing Voice Pitch
9. 3D Animation

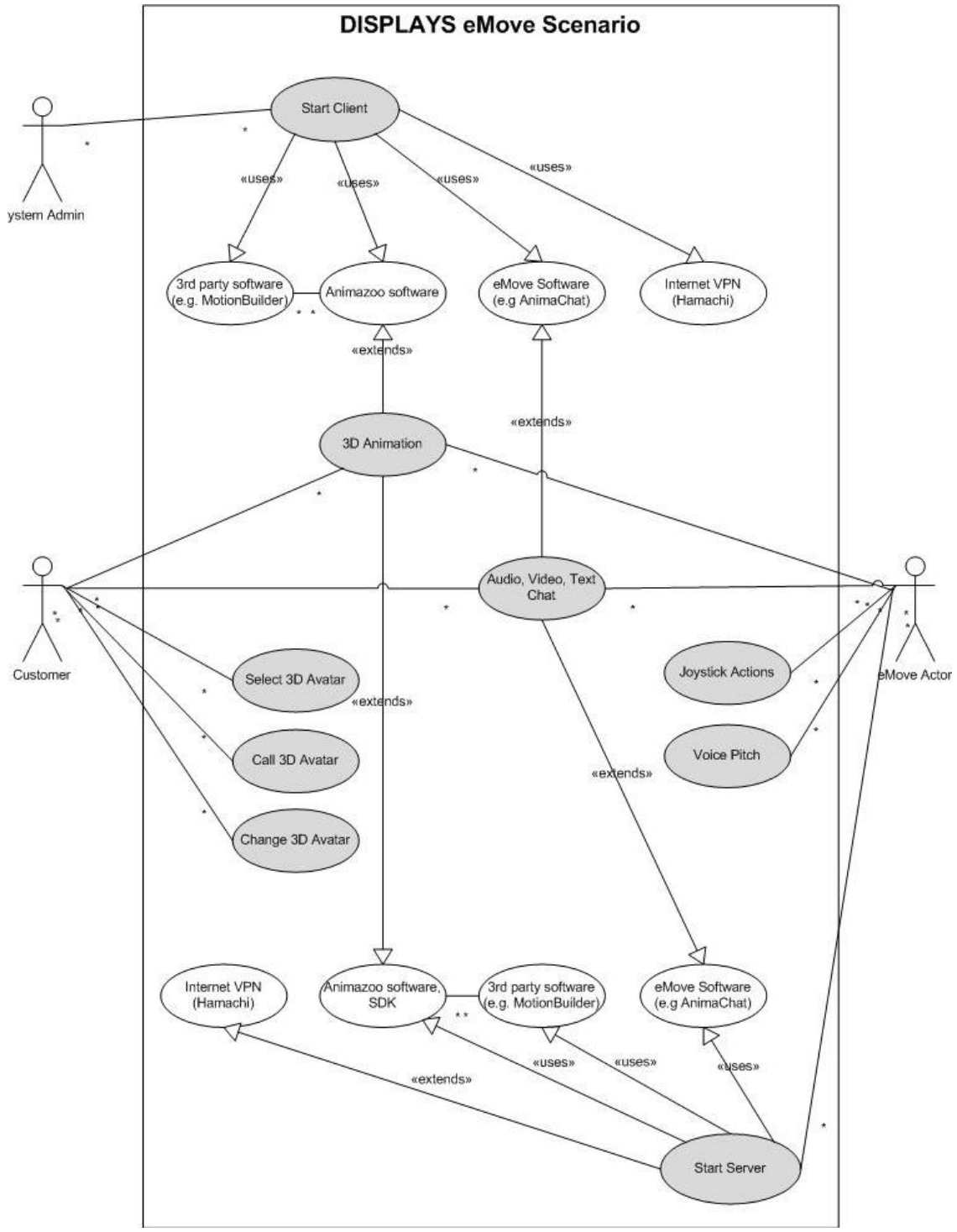

Figure 5.5: Use Case Diagram, Showing Possible Use Cases for Digital Puppetry

5.4.3 Digital Puppet Architecture

Figure 5.6 illustrates the current software digital puppet architecture, together with new proposed software modules and their integration points, needed to implement online digital puppetry application.

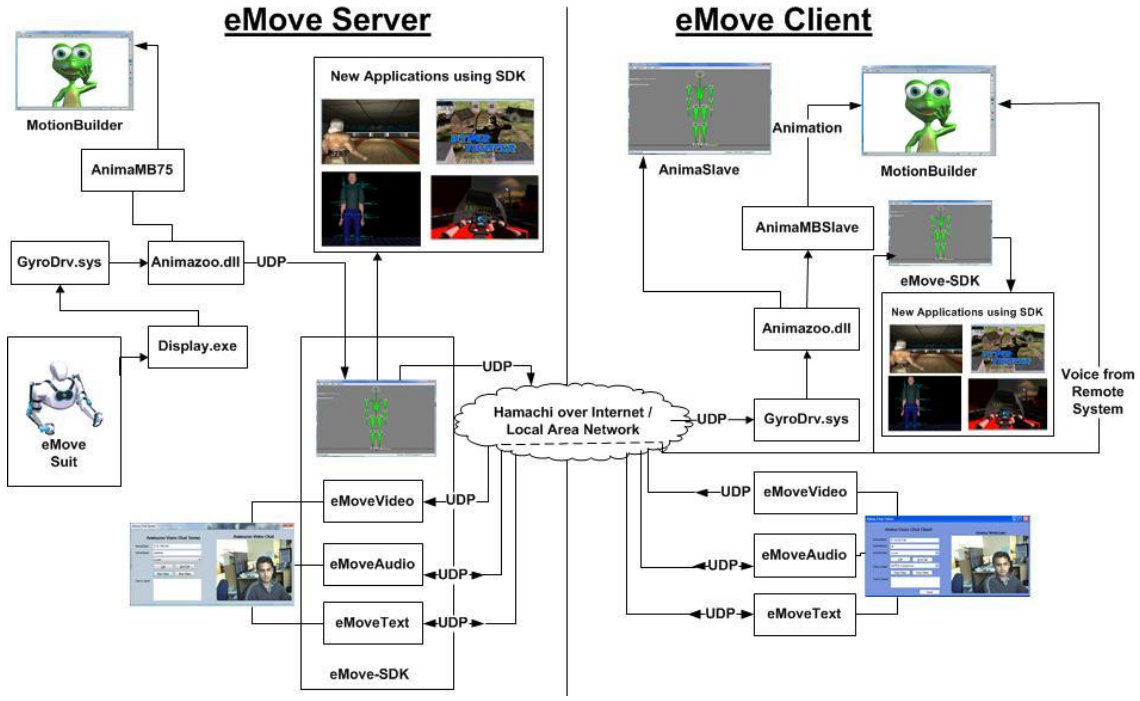

Figure 5.6: Digital puppet system architecture showing various components in client server architecture

From a software perspective, the current development utilises existing Animazoo low level software (AnimaMB75, AnimaMBSlave, GyroDrv.sys, Animazoo.dll and Display.exe) integrated with new software developed in the digital puppet application. This constitutes the innovative online digital puppet system, the architecture of which is illustrated in Figure 5.6. The new software is highlighted with a blue border. Each of the components in Figure 5.6 are described next, then a description of how to implement a digital puppet system is given, followed by a more detailed description of the eMove-chat application (which comprises the components outlined in Figure 5.6 with the blue border).

A. Display.exe

Display.exe handles all the data coming from the motion capture suite. This software can stream the data to AnimaDemo, eMove-SDK and AnimaView software through shared memory.

B. GyroDrv.sys

GyroDrv.sys is Intersense drivers for the orientation tracking devices (such as Intersense Gyros used in both eMove and Gypsy 6 suits).

C. Animazoo.dll

Animazoo.dll is a dynamic link library that processes the raw data into recognisable motion data for each joint.

D. eMove-SDK

eMove SDK is my modified version of the old AnimaDemo, which is an open source software; it shows graphical representation (a 2D avatar) for motion capture data. The AnimaDemo SDK required modification in its UDP packets or an additional UDP layer that can synchronize animation data with voice and keyboard data, and stream it to an end user application (i.e. MotionBuilder, or other games engines). This SDK now includes interfaces to many new applications (e.g. games engines, scientific simulation, etc.). For more information on eMove SDK please see ‘eMove SDK Implementation’, Section 5.5.3.6.

E. Animaview.exe

AnimaView is a compiled version of AnimaDemo (an exe file). It takes the motion data and shows it in a graphical representation based on the .bvh skeleton standards, additionally, it serves UDP packaged data to MotionBuilder or AnimaSlave.exe.

F. AnimaMB75.dll

AnimaMB75 is a plug-in developed for MotionBuilder 7.5 for Animation and Lip-Sync, all on a local machine.

G. AnimaMBSlave

AnimaMBSlave is also an Animazoo plug-in for MotionBuilder but this allows MotionBuilder to get the Animation data from a remote machine rather than a local machine only.

H. Motionbuilder

At present we use MotionBuilder for animation and lip-sync data coming from a remote machine into a 3D avatar inside MotionBuilder using AnimaMBSlave. This third party software can be replaced in the future by any other third party or open source software solution. For this I have tested many solutions such as Panda3D, Unity3 and Unreal game engines. See future work for more details.

I. AnimaVideo

AnimaVideo is part of the eMove-chat functionality that provides video from the eMove client to the eMove server. This functionality allows an eMove actor to see the customer and respond to him according to his/her behaviour.

J. AnimaAudio

AnimaAudio is also an eMove-chat functionality that provides bi-directional audio between client and server. This functionality with dual sound cards inside the system allows the lip-syncing feature inside MotionBuilder.

K. AnimaText

AnimaText is an additional feature of the eMove-chat application that provides text chat and an additional trigger facility for eMove applications. So, if someone needs either text chat functionality or additional triggers, he can use this functionality (currently commented).

5.5 Implementation of the Digital Puppet System

This section describes the implementation details of a digital puppet application for a digital heritage environment, i.e. it describes how to build a digital puppet system from the components discussed above. There are many software and hardware components involved during this implementation so this section deals, on the whole, with those components developed by the author to allow the whole system to be implemented. The implementation section is distributed in various sub-sections. Section 5.5.1 is an introduction to digital puppetry for digital heritage; Section 5.5.2 is about the hardware components involved and their configurations; Section 5.5.3 gives details of software components involved in digital puppetry, their implementation and configurations; and Section 5.5.4 is dedicated to the network infrastructure involved to run the digital puppetry application.

5.5.1 Digital Heritage Puppet Application

The digital puppet application is a network-based application that uses client-server architecture. Although the digital puppetry application has also been designed for a single PC, due to the eMove project requirements and possible heritage requirements for providing the services on multiple screens and at a remote location, it has also been implemented as a client-server application. The client-server version can also operate as a stand-alone version by using ‘AnimaMB75.dll’ instead of ‘AnimaSlaveMB75.dll’, and providing a local host IP address instead of remote machine IP addresses. This application has been mainly tested on local area network and Internet-based wide area networks using ‘AnimaSlaveMB75.dll’. Hence, all the discussions in this research will be based around the remote version of a digital puppet application.

The digital puppet application allows animation (along with trigger events) and audio data streamed online into the 3D avatar’s application program (MotionBuilder in this case) where the avatar interacts with people in the real world. The application server also receives a video of its clients, so that the mocap actor can engage in a meaningful dialogue with its clients.

Figure 5.7 shows digital puppetry client-server architecture with various elements listed. Note: the animation and video data are one-directional, although the audio and text are bi-directional. Mainly there are two software components required for a digital puppetry application.

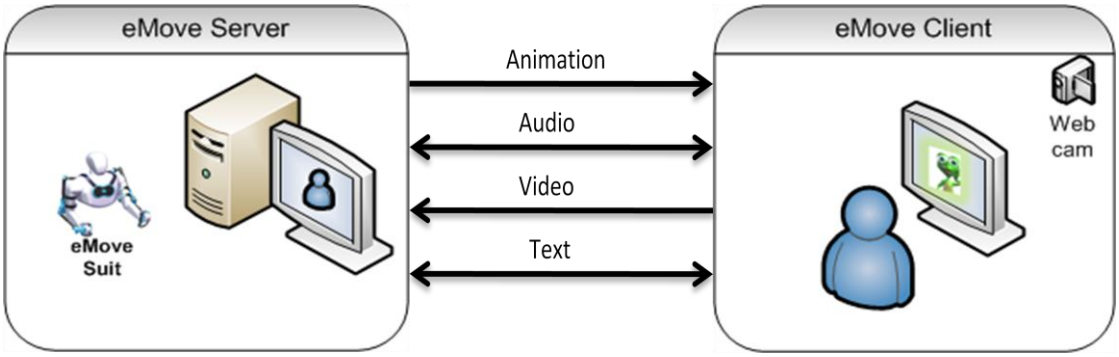

Figure 5.7: digital puppetry for digital heritage: client-server architecture

1. eMove SDK

   - This is a modified version of AnimaDemo with eMove-chat client and server versions with new functionalities (video and audio components) implemented inside the menu titled as 'Chat'.

   - eMove-SDK is used on the server side for streaming animation and audio data to the client machine. It also receives the client's video and displays it on the server machine.

   - The eMove-SDK is also required on the client side for voice conversation and streaming the client's video to the server machine for the eMove puppeteer to see his audience and respond accordingly.

2. MotionBuilder

   - This is used on the client machine for an animated 3D avatar on the client screen, through the real-time motion capture and voice data streamed from the eMove server.

5.5.2 Hardware resources

The hardware resources include:

- Two personal computers
- One motion capture suit
- Two headphones with built-in microphone, alternatively two speakers with two separate microphones
- One external soundcard (not required if the client machine already has two soundcards).

5.5.2.1 *Computing resources*

Typically, running a Gypsy 6 suit (the previous version of an eMove suit) requires a Pentium–II 450Mhz processor with 256MB memory, a decent video graphics card, and Windows NT/ 2000/ XP / Vista 32 bit operating systems. During this work, it has been observed that running an SDK for streaming the data inside other programs needs more memory and computing resources. Also the 3D graphics applications require a much higher PC configuration such as the MotionBuilder require Pentium–IV or higher machine with minimum 1GB memory and 128 bit graphics card.

Based on MotionBuilder 2009 and modified Animazoo SDK, this work recommends a computer system should have the following hardware configuration.

- Processor: minimum Pentium – IV processor / equivalent or recommended Duo core processor or higher.
- Memory: minimum 1GB or recommended 2GB
- Graphics card: minimum 512MB or recommended 1GB
- Operating system: Windows 32bit XP or VISTA

5.5.2.2 *Dual Soundcards for client PC*

The MotionBuilder software by default is set to hear the voice from the microphone of the attached soundcard, i.e. it hears the locally attached microphone but not the remote microphone. The networked version of a digital puppetry application requires the sound

of the remote machine fed into the avatars lips. Thus, to enable the system to receive the voice from a remote machine, two audio channels are required in the client machine. The audience's voice has to be isolated from the MotionBuilder avatar. This requires a sophisticated sound card with two channels, and perhaps software as well. However, an innovative trick is simply to use two sound cards in the client machine and configure them as described below.

Figure 5.8 shows the second soundcard that can be inserted in the machine's USB port.

Figure 5.8: Creative Sound Blaster X-Fi Go!

The client machine needs to be set up as follows:

- Attach USB soundcard into USB port of client computer and install drivers
- Insert microphone jack inside microphone input of USB sound card
- Insert headphone or Speaker Jack into speaker port of built-in soundcard

5.5.2.3 *eMove Motion Capture Suit*

An upper body eMove motion capture suit is required for implementing a digital puppetry application. Figure 5.9 (a) shows the development version of the eMove motion capture suit used during the development phase of this work, and (b) is a CAD design of the final version of the eMove suit. Note: these suits are almost identical to each other with similar hardware and software configurations; the only differences for the new eMove suite are that it is lightweight and has a new improved design.

Figure 5.9: (a) Author and developer of this work wearing development version of eMove motion capture suit. (b) CAD design for final version of eMove suit to be released by mid-2011

The motion capture suit consists of sensors for chest, shoulders, elbows and hands based on a hybrid electromechanical and inertial gyroscopic sensor approach, with two joysticks to cover anything else we might need (such as walking, stepping up, sitting and triggering finger actions). The eMove motion capture system is scheduled for launch in April 2011 at a consumer level price of around $300 targeted at the online gaming market. This revolutionary new system is the foundation stone for one of the main innovations in DISPLAYS heritage interaction and virtual walkthrough.

The Gypsy 6 upper body motion capture suit used for digital puppetry research, which is very similar to the eMove motion capture suite with built with:

- 16 Potentiometers (for both arms)
- 2 Gyroscopes (1 for head and 1 for chest),
- 2 Joysticks (with three keys in each giving total six trigger events)
- 1 PCB board (to calculate the rotation information)

5.5.3 Software resources

The main software components required for digital puppetry application for digital heritage are:

- eMove suit drivers
- eMove SDK (modified AnimaDemo)
- AnimaSlaveMB
- MotionBuilder
- eMove-chat application

5.5.3.1 *eMove Suit drivers*

The eMove suit driver is the same as the Animazoo suit driver as already discussed above: the eMove mocap suit is identical to Animazoo's Gypsy 6 suit. The driver suit consists of many components including Display.exe, AnimaView.exe, AnimaSlave.exe, Animazoo.dll and install.bat files (already described above in the 'System Architecture' section of this chapter, except the install.bat file). The install.bat is an installation file to copy the mocap driver files and Intersense gyro driver in a user's computer (system32 folder), and makes entries into the system registry. These files and entries are required in order to work with a motion capture suit.

Once the driver is installed, the Display.exe file can be initiated, see Figure 5.10, display.exe file showing raw channel data.

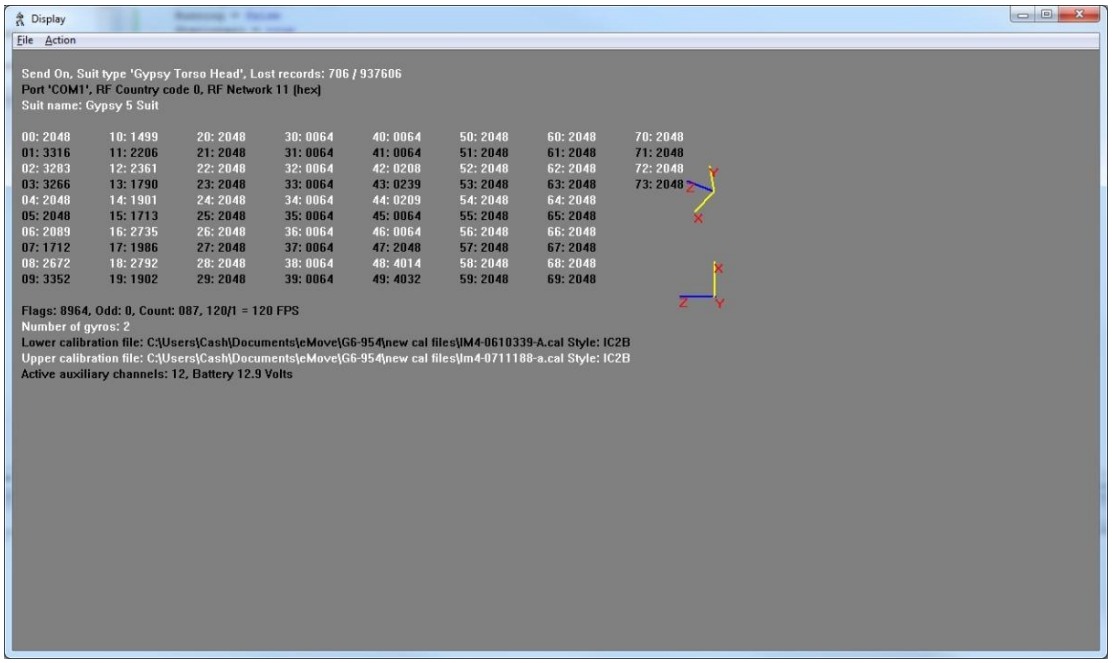

Figure 5.10: Display.exe file showing mocap raw channels data

After Display.exe is loaded showing the channel data, this data can then be streamed to either AnimaView.exe or eMove SDK. To stream the data into either of these, click ‘Actions’ Menue=> ‘Send Packets’, this will open a network port and wait for incoming connections.

5.5.3.2 *eMove SDK/ Modified AnimaDemo*

The eMove SDK was modified from the AnimaDemo version with much additional functionality (including eMove-chat, real-time motion capture data stream to different games engines) all implemented as a part of the digital puppetry application for digital heritage.

Once the server in display.exe is connected, the eMove SDK / AnimaView.exe can then be initiated. Figure 5.11 shows the eMove SDK connected with real-time mocap data streamed from Display.exe.

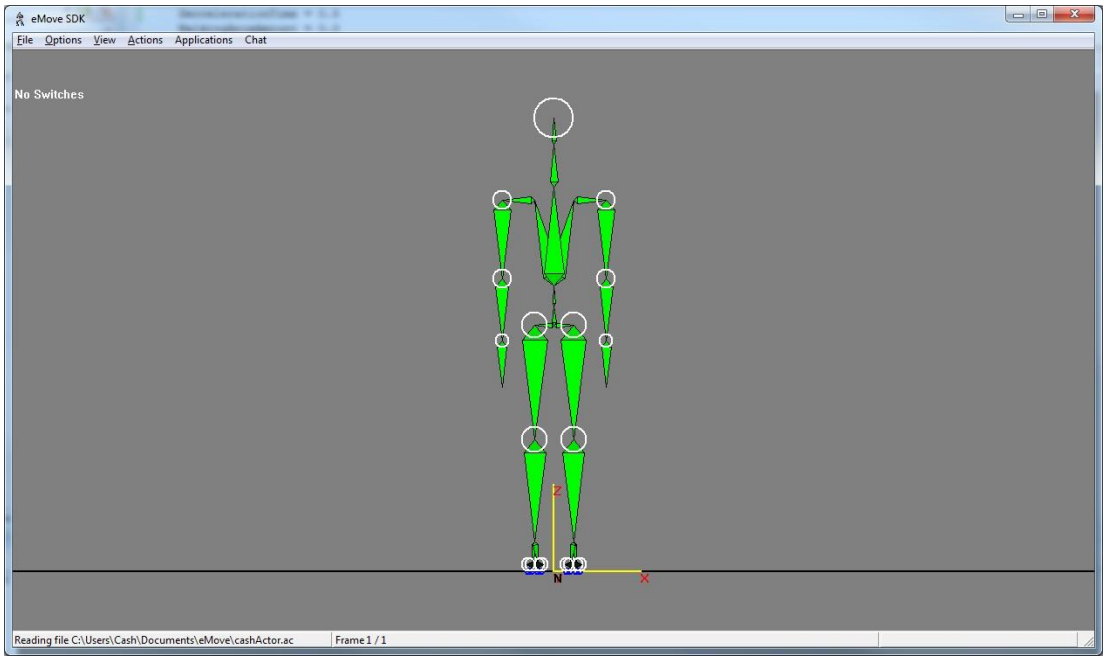

Figure 5.11: eMove SDK connected with motion capture data through Display.exe

Once eMove SDK is running and the skeleton inside the SDK window is moveable, it is time to stream the animation data into MotionBuilder or other suitable game engine. This can be done by enabling the server from eMove SDK and then connecting AnimaSlaveMB from MotionBuilder. To enable the server inside eMove SDK, click 'Actions' Menue => click 'Enable Server' (Alternatively press Alt+S buttons), a small dialogue box will appear, see Figure 5.12, an IP selector to choose between different LAN cards / IP addresses, this automatically loads all IP addresses assigned to the current system, and includes physical as well as virtual IP addresses.

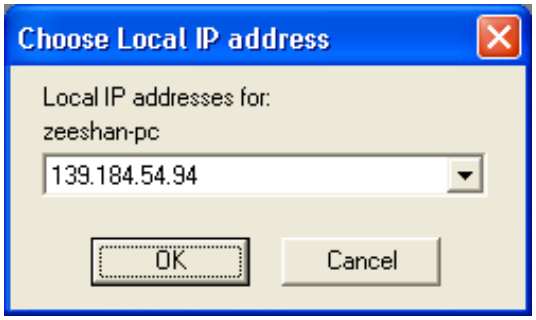

Figure 5.12: The IP selector dialogue box for 'Enable server' option inside eMove SDK

In this case the local IP address is selected but in case the client machine is at a remote location (i.e. not in the same building or local area network and connected through the

Internet), the user needs to select a virtual IP address to stream through a virtual private network.

5.5.3.3 *AnimaSlaveMB*

AnimaSlaveMB is Animazoo's MotionBuilder driver to accept the mocap data from a remote PC through AnimaView or eMove SDK software. The AnimaSlaveMB is a DLL file that needs to be launched inside MotionBuilder, which provides several options such as local and remote IP addresses, connect/disconnect buttons, live, record and play options. The user needs to provide both IP addresses and press 'connect' to establish the connection with the server (pre-conditions eMove SDK 'server option' should be enabled, and the remote IP should be the same as the eMove SDK server IP address). This will connect MotionBuilder to eMove SDK of the remote machine and present a stick character (a skeleton), moving exactly as in eMove SDK.

Once MotionBuilder is connected, a character can then be imported and merged with a skeleton using the biped option. This will allow the character to follow the skeleton and, as the skeleton data is real-time data from eMove SDK, this makes the character inside MotionBuilder follow the movements of the human connected by the remote machine.

Figure 5.13 illustrates an avatar (clearly not the Roman Emperor in the scenario above, but for illustrative purposes this character is animated and talking, being driven by an eMove suit) inside the MotionBuilder 3D environment connected with real-time motion capture data through eMove SDK. Once the avatar is connected and moving through real-time mocap data, it is easy to use the eMove-chat application and stream audio data inside the avatar's lips to perform lip-syncing. The next section describes this concept using the eMove-chat application.

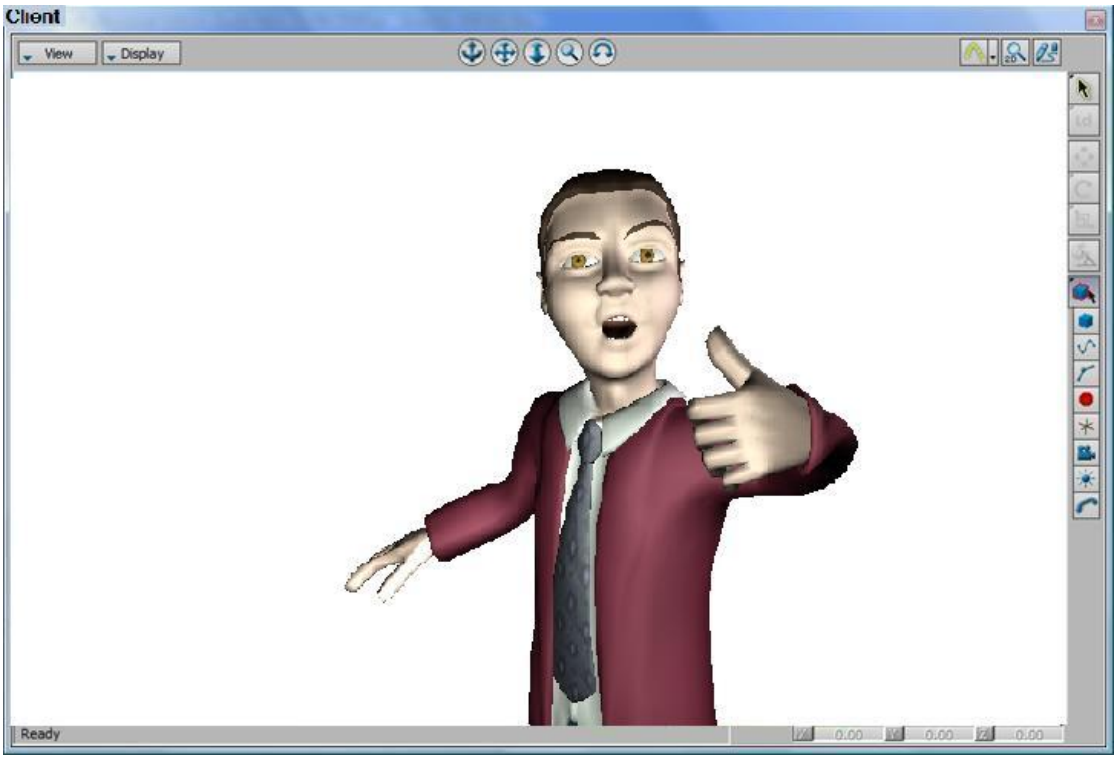

Figure 5.13: A character inside MotionBuilder connected with remote machine through eMove SDK (Real-time Motion capture data)

5.5.3.4 *eMove-chat Application*

The eMove-chat is a client-server chat application developed as a part of this work to enable the digital puppet to speak through the voice streamed from machine located in either a remote location or in the same local area network. The eMove-chat is an Internet-based UDP audio, video and text chat application that allows the person wearing the eMove suit to communicate and see the audience. His voice is captured through a normal microphone and streamed through an Internet/local network connection into the client's PC, and is fed to a 3D avatar in real-time. The person wearing the eMove suit can see his audience through the web camera attached to the client's computer. He can also chat with him through the text chat facility whenever required.

Although the eMove-chat is an independent application, it can also be invoked through the 'eMove-chat' option available in 'Chat' menu in eMove SDK. This can be loaded from either side (client/server); it does not make any difference. Once it is loaded enter

the IP address of the other side and press call, the other party will hear a decent telephone ring tone and a dialogue box, will pop up with the options to either 'Accept' or 'Reject' the call. On 'Rejection' the call originator is informed with a busy tone and notification dialogue box and on acceptance of the call a voice session is established between both parties.

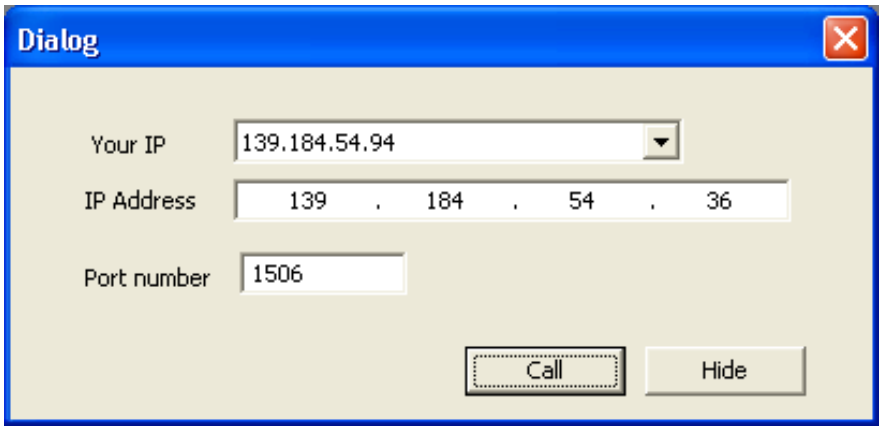

Figure 5.14: eMove voice chat call dialogue box, available in 'Chat' menus in eMove SDK

The eMove client can start his camera for the person in mocap suit to see him. Once the client has started his camera, the digital puppeteer can enter the client's IP address, press 'connect' and view his video.

Figure 5.15: eMove video chat (server side) window to view remote (client) camera

Figure 5.14 shows the eMove audio chat dialogue box, and Figure 5.15 presents the eMove video chat dialogue box. Note: the audio dialogue box is similar for both client and server applications, though the video dialogue box shown above is the server side view of the eMove-chat application. The client side view is shown in Figure 5.16 below with MotionBuilder. The eMove server can see his client's video and the client can see a 3D digital avatar moving and acting instead of the person's video. See Figure 5.16 below for a visual description.

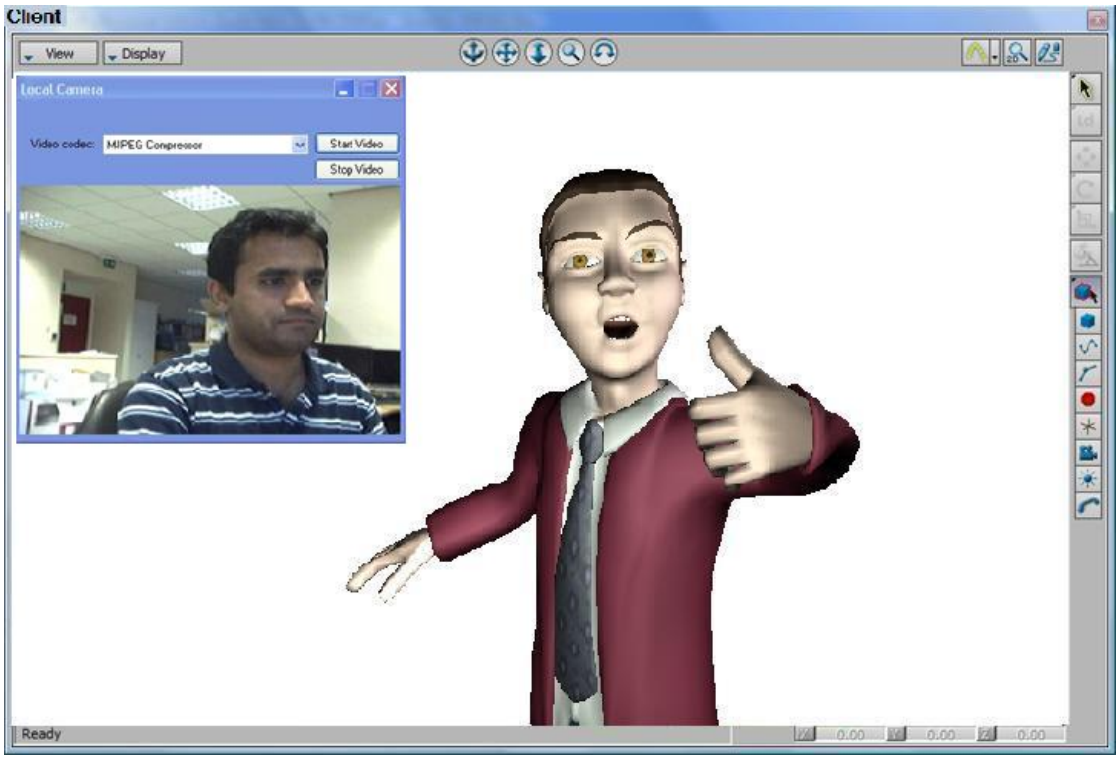

Figure 5.16: Digital Puppetry Client Side View

eMove video component, displaying client's own video, so the client can see that he is being viewed by puppeteer. The eMove video chat on top of MotionBuilder both connected to eMove server (digital puppeteer's machine)

5.5.3.5 *Lip-syncing and Facial Expression*

The lip movement and facial expression are a very important part in digital puppetry to demonstrate the past in present. The lip movement is synced with audio coming from the remote machine streamed inside MotionBuilder, and the facial expressions are blended motions controlled by the switches available on the eMove mocap suit

(joysticks) on the remote machine. The eMove SDK is also responsible for streaming the triggers data along with the mocap data from the eMove server machine to the client machine. The facial expressions include: smile, laugh, wink, eye blinking, and sadness, all controlled by five joystick buttons.

Although the default settings inside MotionBuilder accepts only the voice from the microphone from the local machine where it is installed, the two soundcards configuration explained above in Section 5.5.2.2 allows MotionBuilder to detect a voice from the remote PC instead of the local machine.

5.5.3.6 *eMove SDK components*

The eMove SDK includes a new version of AnimaDemo and the innovative new eMove-chat application, and other software interfaces useful for many other applications namely:

- Panda3D motion capture driver
- Unity3D motion capture driver
- Unreal motion capture driver
- Supercollider programming language motion capture driver
- Siemens Jack software motion capture driver
- Java based gesture recognition system motion capture driver

The implementation of eMove-chat, and those applications such as the game engine drivers for Panda3D, Unity and Unreal, allows others to develop applications such as the digital puppet for heritage use. Example applications developed with the additional so far are illustrated in Figure 5.17.

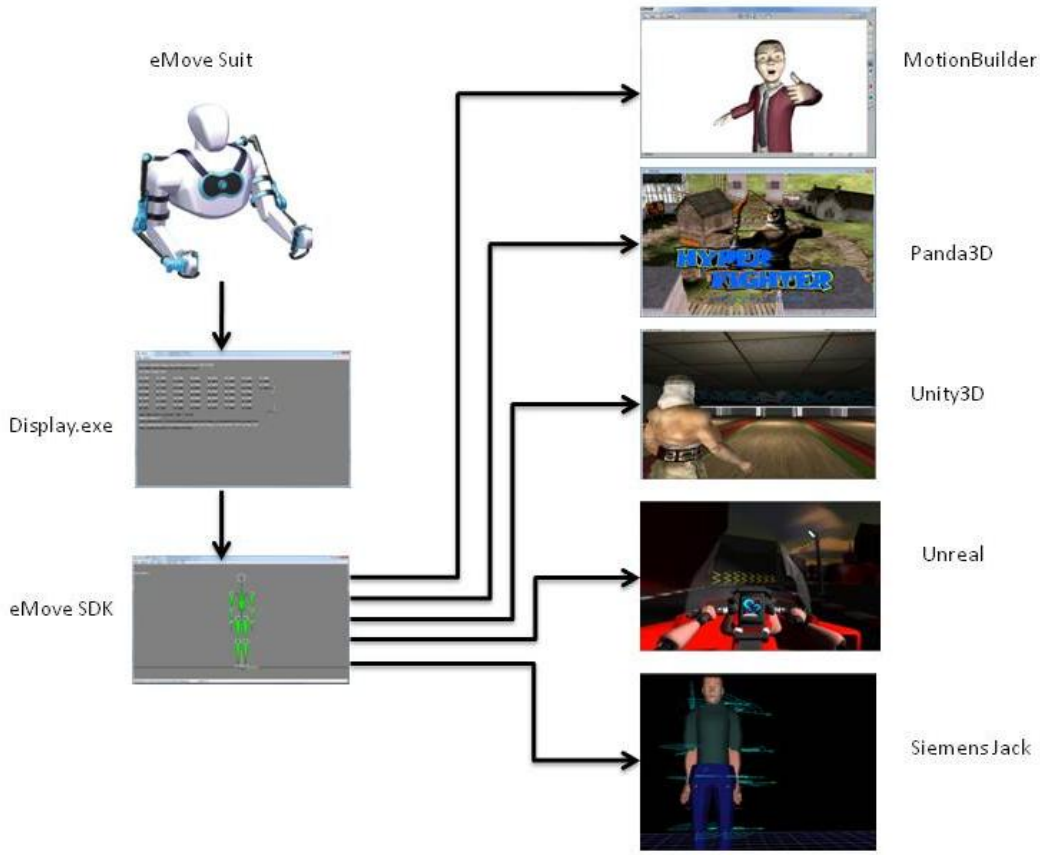

Figure 5.17: Mocap data flow diagram from eMove SDK to game engines

Specifically, the interfaces to games engines (Panda3D, Unity3D and Unreal) are developed to reduce the cost of digital puppetry, particularly to eliminate MotionBuilder (and allow the creation of online gaming markets, but that is another story beyond the scope of this thesis). This was already mentioned in Section 5.1.1, but worth mentioning again because the implementation of the motion capture game engine drivers as an innovation opens up new opportunities for exploiting such technology in a digital heritage system. What Figure 5.17 shows is that the innovative eMove motion capture technologies (suit and SDK) opens up new opportunities for the digital economy, in particular it shows several examples implemented that could be adapted for heritage scenarios: the digital puppet based heritage character, a battle scene between ancient warriors (hyper fighter), the next one is a bit crazy with the ancient character in a modern bowling alley, but the point illustrated is the versatility of connecting in real-time motion capture data to a modern gaming engine. This is a first!

One of the innovative components of the eMove SDK, apart from the game engine drivers, is the eMove-chat application that is a major component of the digital puppet application, whose implementation details are discussed next.

5.5.3.7 *eMove-chat Implementation Details*

The eMove-chat application is a network-based application based on client-server architecture. The current development of eMove-chat has separate audio, video and text chat components all available inside eMove SDK menus. The communication between client and server is one-to-one bi-directional for audio and text, and one directional client to server for video (note: video can also be bi-directional for other applications, where required). The client and server can communicate with each other on either a local area network (LAN) or Virtual Private Network (VPN).

A. Voice over IP

The requirement for a digital puppetry application is to stream online (both Internet and local area network) motion capture data along with voice data in real-time; hence a voice-over IP solution was required. Many voice-over IP (VoIP) systems have been developed, for example, free online chat programs such as Yahoo, Google, Windows Live Messenger and Skype are leading names in voice-over IP technologies.

The question arises: Why a new voice-over IP solution was built as part of this research when there are many successful applications available on the Internet?

There are four main reasons to build a new solution rather than using one of the existing solutions:

1. The existing VoIP applications provide low quality of service;
2. They either work on the Internet or local area network, they do not work on both;
3. The commercial solutions are not customizable for branding and further development;

4. Synchronization with animation data; the next step in our solution would be to integrate both voice and animation data into a single packet to compare synchronization between two systems.

None of the existing VoIP applications support animation streaming, although some provide an SDK, e.g. Skype. However, after examining the Skype SDK it was realised that it only allows modification to the interface or presentation layer of the application; also there is no chance to build an LAN-based application out of Skype SDK. Also later on modifications to the underlying communication code may be needed, and new code added for synchronization between animation stream with audio and text data. This may involve adjusting the packet size, bit rate, or may be different codec, etc., to improve quality of service. When running this application on the Internet, we use a virtual private network solution that requires logging to a server, but when this solution is deployed on an LAN environment, we do not need to connect to the server, in fact sometimes there may not even be an Internet connection.

A new application has been already developed with total control on the developed application for its further modification, starting from the basic networking using UDP protocols; this is useful for adding additional application layers as the application demands. For example, this solution can be customized for security using additional layers on top of UDP packets, the quality of service can be improved using different encoders, sample rate, and variable packet size, or even additional layers of RTP (Real-time Transmission Protocol) can be added on top of the UDP packet to acknowledge the packets drop.

B. Audio Codec

The voice-over IP application required the choice of an appropriate audio codec, preferably open source, but a cost effective commercial codec would also be viable. There are many open source audio codec available to implement inside existing or new applications.

Most of the voice-over IP Internet applications based on the UDP communication method suffer from voice packet loss, hence it is a universal fact that UDP is a

connectionless protocol (i.e. data packets are not guaranteed to arrive). Nevertheless UDP is recognised as the fastest networking protocol best suited for streaming data by many online multimedia streaming applications. Other main factors affecting the voice quality include the choice of codec, echo control, packet loss, delay, delay variation (jitter), and the design of the network. UDP-based communication VoIP applications can be improved by careful studies and adjustments of these factors [165].

The eMove-chat application development was started using a popular open source voice codec named as G.711, also called A Law (a-Law) when used in European PCM systems, and U Law (μ-Law) used in American PCM systems. These are the basic laws for implementation of voice encoders and decoders [166]. The G.711 gives good performance on a local area network but observed poor performance was observed on Internet-based tests using the Hamachi network. No doubt one of the reasons would be that the VPN networks do not utilize the whole Internet bandwidth. Of course, one can buy a dedicated Internet line with static IP addresses on both ends to improve the performance, but there were financial limitations that involved using public IP addresses, and the only way found was to build a P2P tunnel between two computers using a kind of virtual network. Using the G.711 protocol, delay was experienced over Internet-based connections and some voice breakages. After deeper studies on voice-over IP solutions, various other codec were found, such as Free Lossless Audio Codec (FLAC) [167], PortAudio [168], Speex [168] and iLBC (Internet Low Bitrate Codec) [169].

Having carried out enough background research, the Speex encoder and decoder have been implemented inside the eMove-chat application. Speex is also an open source implemented C language provides excellent sound quality on receivers end. The Speex codec works on many bands 8, 16 and 32 KHz, and different sample rates.

The configuration that worked well for my application on both the Internet and local area networks was channel 1 (stereo sound), sample rate 8k with 16 bit per sample. See below a code snippet.

```
#define CHNL = 1; //Sterio
#define SAMPLE_RATE (8000)
#define int BIT_PER_SMPL = 16;
```

The eMove voice chat application works well between various points except when the broadband connection is shared between many users; then voice chat breakages are experienced. However the eMove voice chat works fine on a local area network, and that is the requirement for our digital puppetry application for digital heritage environments.

C. Video CapX

For real-time video streaming there are many open source, freeware and commercial API and tools available. The eMove project funded the purchase of a small commercial API instead of 'reinventing the wheel', a cheap commercial API named as VideoCapX [170] was used that allowed easy integration of video components inside both client and server sides of the eMove-chat application.

The VideoCapX API also provides client and server side components that can be integrated by calling some function call statements. However the eMove server needs to see a client's video, hence it needs the VideoCapX client component; similarly the eMove client needs the VideoCapX server component to broadcast a client's video. The VideoCapX client has been added to the eMove-chat server side and the VideoCapX server component has been added to the eMove-chat client.

5.5.4 Network Architecture

As stated above, the digital puppetry application for digital heritage can be stand-alone and networked based. The network-based application can work on both LAN and WAN through an Internet connection. There are two ways to implement the P2P connection between two points over the Internet:

1. Static IP address;
2. Virtual Static IP address.

The first one (static IP addresses) is quite expensive and not everyone can afford it. The second one is a cheap, possibly a free solution, where the two end points are assigned a virtual IP address in addition to a physical IP address. The virtual private network (VPN) was used during these experiments illustrate the online real-time digital puppetry concept: see more details below.

5.5.4.1 *Virtual Private Networks*

A virtual private network (VPN) is an isolated and private network tunnel between two end points over any public network. This allows network data to be sent independent of other network traffic [171] . The eMove-chat client and server application is designed to work in socket-based IP networks, which can either be an IP addresses from a local area network or an IP address from an Internet-based virtual private network. The eMove-chat works in a similar way in both cases, the user just needs to specify the IP address in the dialogue box provided.

The eMove-chat application may be required to work on an Internet-based wide area network to run a digital puppetry application. This means that a point-to-point connection between client and server machines needs to be established, where mocap data, audio, video and text data can be streamed without any interference. The static IP addresses are expensive but the virtual private networks are ideal in this type of scenario.

There are many open source, freeware and commercial VPN solutions available on the Internet that can be adopted as part of a digital puppetry application. For digital heritage environments where expert technical computer network people may be difficult to find (to fix the network whenever there is any issue), a very simple configurable solution that could be easily installed and configured by the end user needed to be found. It should also be independent of physical IP addresses so that it can be moved anywhere in the world for demonstrations or any other purposes.

Open source solutions such as OpenVPN [172] can be a good solution for establishing a VPN-based tunnel, but it involves working with command prompts and some other configurations such as manually specifying the server address, encryption and

decryptions methods, authentication, etc.; this solution would be difficult for a new user. A personalized VPN using Windows built-in VPN functionalities could be made, but this would still require sufficient technical knowledge during installation and set up, and the end-users may not be able to do it on their own after every new installation.

Currently the digital puppetry application uses a freeware version of Hamachi [173] virtual private network solution. The Hamachi is an easy to use and easy to configure commercial and shareware solution; it is also called a zero-configuration solution for developing virtual private networks between computers behind firewalls. A freeware version of VPN software such as Hamachi is an ideal solution to demonstrate the concept and perhaps it can do a good job for a network with only a few computers. A free version of Hamachi can be used to connect eight computers in a network, and in our current solution I have only two computers to be connected, hence this is a perfect solution at this time.

5.6 Testing and Evaluation

The digital puppet application (i.e. the 'Interaction using motion capture' for a heritage characters scenario) for a digital heritage application has been tested on various occasions and between various places. It has been tested on a local area network as well as on wide area network based on an Internet connection using Hamachi VPN software.

In a local area network the digital puppetry alongside the eMove-chat application works perfectly well, it has even been tested over the Internet connection between two buildings in the same city (Brighton). What this illustrates is that a remote digital puppet application (one that broadcasts or multicasts the avatar to or from a studio in one location in a theme park or museum to remote kiosks around the theme park or museum) are viable because they will have secure intranets with sufficient network bandwidth.

It provides a good performance without any significant delay for audio, video or animation data transfer. However, the wide area Internet-based puppetry tests (which might not be required for digital puppetry), initially had many issues with voice breakages and UDP packet drops. However, after some more research on VOIP

applications, performance for short sessions was improved to a reasonable level. By replacing the G.711 codec with Speex codec, and reducing the sample rate from 16k to 8k has significantly improved the eMove-chat performance in this respect.

Nevertheless, to produce an effective production system that, for example, implements an 'Interaction through motion capture' using heritage characters, where an ancient person tells the story of the Parthenon from a digital puppet system installed in Athens to a group of visitors to the British Museum looking at a kiosk next to the Elgin marbles, will require access to an 'efficient' network bandwidth for the system to operate over a long period. The digital puppetry application has been tested between following places:

Table 5.1: shows the Digital puppetry tests results on the scale of Worst to Excellent

Location-1	**Location-2**	**Distance**	**Audio**	**Video**	**Animation**
CGCLab, Sussex	CGCLab, Sussex	Short: Same Building	Excellent 5	Excellent 5	Excellent 5
CG & IS lab, Sussex,	Animazoo, UK	Short: Same City	Good 4	Good 4	Excellent 5
CG & IS lab, Sussex	Lass Vegas, USA	Long: UK to USA	Fair 2	Fair 2	Good 3
CG & IS lab, Sussex	New York	Long: UK To USA	Good 4	Good 4	Excellent 5
CG & IS lab, Sussex	Pakistan	Long: UK To PK	Fair 2	Fair 2	Fair 2
CG & IS lab, Sussex	Hong Kong	Long: UK To China	Fair 2	Fair 2	Fair 2
CG & IS lab, Sussex	Moscow, Russia	Long: UK to Russia	Fail 0	Fail 0	Fail 0

CG & IS lab, Sussex	California, USA	Long: UK To USA	Good 4	Good 4	Excellent 5

Scale definition:

Excellent (5) → worked perfectly fine, throughout the test, with no packet drops.

Good (3 – 4) → Occasionally few packets drop with few glitches but sound still clear, video and animation was fine, and understandable.

Fair (1 – 2) →significant packet drops, due to a poor internet connection on other side , but voice still understandable but poor quality, animation worked fine, and video with dalay of 2 to 3 seconds.

Fail (0) →Didn't work at all

Abbreviations:

CG & IS: Interactive Systems: Computer Graphics Lab

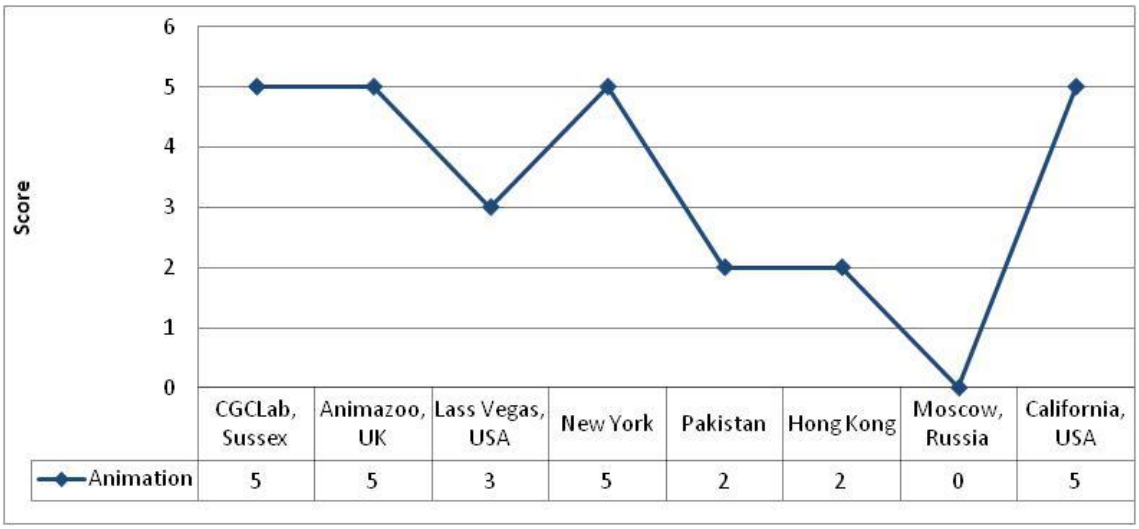

Figure 5.18: A graph showing score for test results for animation data between Brighton, UK and other places

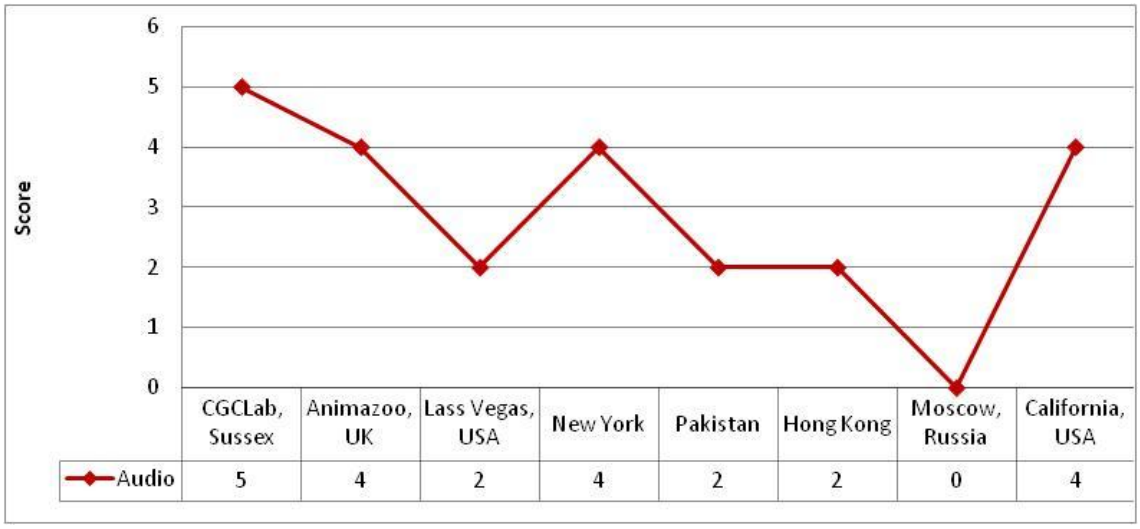

Figure 5.19: A graph showing score for test results for audio data between Brighton, UK and other places

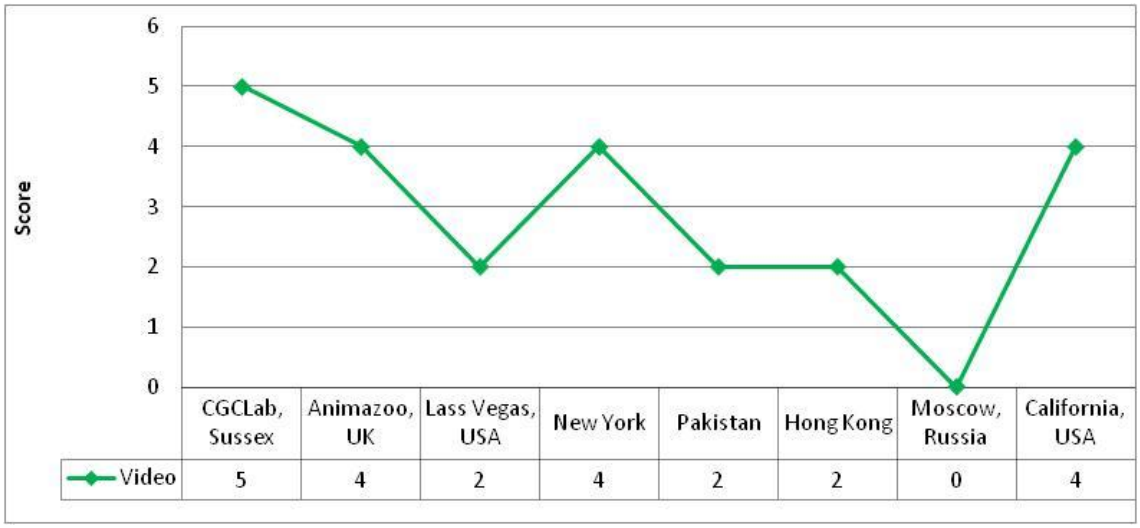

Figure 5.20: A graph showing score for test results for video data between Brighton, UK and other places

Table 5.1 shows various networks tests and their results implied during this research on a digital puppetry application. The tests are in descending order with the most recent on top and the oldest at the bottom. The tests between Moscow-Brighton and Las Vegas-Brighton were held using a poor and shared Internet connection provided during expo's, which did not allow the digital puppetry to operate properly. Figure 5.18, Figure 5.19 and Figure 5.20 visualizes the same results in separate graphs with scores for each destination.

5.7 Summary

Innovative Motion capture based interaction techniques presented in this chapter are unique for digital heritage systems. The motion capture based real-time interaction, along with audio and video capabilities, has been tested for use within a DISPLAYS type of system. There are three ideas (scenarios) discussed in this chapter. The main idea is to produce an online digital puppet system for digital heritage, where the digital puppeteer can narrate a story of the past to museum visitors (inside museums) and other related community members (elsewhere in the world). The other possibilities include allowing museum visitors to interact with a digital heritage environment and objects in real-time using motion capture drivers for various games engines.

This research uses the eMove motion capture suit, eMove SDK and eMove-chat applications to demonstrate the digital heritage puppet concept, which can easily be deployed inside any museum with heritage assets and models. The innovative eMove-chat application has been built and integrated inside eMove-SDK to allow a digital puppeteer to control digital puppet from remote locations. The digital puppet system has been tested between various remote locations (same city, different cities and across countries), and is satisfactory except in the case of low bandwidth where the UDP packets are likely to drop sometimes. However, this limitation was observed when the test was carried on a heavily shared bandwidth, such as at a conference or expo, where hundreds of users share the same Internet connection.

This work also includes the development of unique real-time motion capture drivers for games engines (such as Unity3D and Panda3D). Such drivers are a first, and enable the creation of new digital heritage systems that deploy motion capture technologies with games engines to visualise and interact with digital heritage content.

CHAPTER VI

6 Conclusion and Future Work

Existing digital heritage libraries or systems are designed to digitise and archive digital heritage objects and metadata elements. Most of these digital heritage systems are collections of 2D images with associated metadata elements; some, e.g. those coming out of particular EU research projects incorporate 3D to visualize a small percentage of selected heritage objects with respect to the total collection. The presentation and interaction parts of these digital heritage systems (a typical system, for example, would be a museum catalogue system) are lacking in many useful and interesting functionalities, which could make these museums, sites and monuments a greater attraction for the public.

The research presented in this thesis was built around the DISPLAYS conceptual framework, as such many components and many different architectural methods could be used to build a complete DISPLAYS system. Implementation of a complete DISPLAYS system is beyond the scope of this thesis as an individual endeavour; other research team members are building other components, including the 'validation architecture': the reanimating cultural heritage digital repository.

The purpose of defining DISPLAYS was to provide a framework upon which to develop innovations in digital heritage technologies for its sustainability, preservation, visualization and interaction. There are five main services proposed for the DISPLAYS framework:

1. DCC: Digital Content Creation Services

2. DCA: Digital Library Archival Services
3. DCE: Digital Library Exposition Services
4. DCP: Digital Library Presentation Services
5. DCI: Digital Library Interaction Services

The research in this thesis was focused on innovations in DCC and DCI services, i.e. content creation, visualization and interaction services for DISPLAYS. The main objectives were to contribute to and implement innovations in:

- The DISPLAYS frame itself as a concept for the implementation of digital heritage systems [6] [7] [117] ;
- A cost effective render grid for use in creating digital heritage content [114] [113];
- A real-time digital puppetry solution for digital heritage character scenarios [119] [120].

The idea was to facilitate a digital heritage community with a cost effective rendering solution for creating digital heritage simulations (e.g. pre-defined walk-through) requiring many thousands of 3D images (frames) for use in a heritage movie. This digital heritage content can then be used for downstream presentation and interaction services. Given the creation of digital heritage content in the context of scenes using DCC services, such content can then be implemented in presentation systems such as gaming engines configured for a particular heritage scenario. What is then missing is the inclusion of real-life avatars that can represent characters of the age who can narrate the heritage story from within the digital heritage environment (virtual world in the game engine). The digital puppetry solutions presented in this thesis pave the way for this scenario, and the visitor can even use the same digital puppet technology to immerse themselves through ‘embodied interaction’ into the heritage world. The above objectives required work on the implementation of a render grid solution and real-time digital puppetry solution using motion capture, voice and video streaming technologies: the research work presented in this thesis meets these objectives.

The main research results for this thesis are summarised separately for each objective, followed by a comprehensive discussion on future work.

6.1 Cost Effective Render Grid Solution

During this research a cost effective rendering solution based on grid computing was implemented and tested. This rendering solution involved the implementation of a grid computing infrastructure based on a Condor high through put computing environment, and a render grid solution implemented in the Python language.

6.1.1 Implementation of Grid Computing

The grid computing hardware infrastructure was developed using old discarded machines to prove the green way of using wasted machine power. The hardware equipments (i.e., 6x Slave/Server machines, 1x Master, and a Gigabit switch) used during this set up, all were old and discarded machinery.

The software architecture was based on a Condor high throughput computing environment, which provides the way for CPU scavenging to use the wasted CPU cycles. Although the rendering setup could be built using a cluster based solution rather than a grid solution, cluster computing will not allow us to use already in-use machines (such as personal computers). To give an example, the Computer Graphics Centre at the time the work was implemented had about 20 in-use research machines (researchers using their PCs, some researchers had more than one PC on the network), a cluster of 10 PCs, various academic office machines, and access to a batch of about 20 very old machines about to be scrapped. This level of computing resource might be similar to a medium sized museum's computing resource. If only a cluster was used for the render grid, a large percentage of available resource would be lost. However, in practice, from a research perspective (disturbing other researchers during development work, etc.) the render grid was proved on a batch of the 'about to be scrapped' machines, see Figure 4.4

6.1.2 Render grid solution based on Grid computing

The application built for an innovative grid-based rendering solution provides seven services. These services are named as:

1. Job Registry
2. Chop Animation
3. Job Submission
4. Render Grid
5. Job Monitor
6. Retrieve Results
7. Encode Results

Specifically, these functionalities provide a complete system management tool to control the whole rendering process. Briefly, a 'job registry' service is used to record new job entries for 3D animations or graphics; the 'chop animation' service is used to build a new grid job for an already registered animation or model; the 'job submission' is used for submitting a created job on a grid computing environment; the 'render job' is used to execute it on a grid; the 'job monitoring' is used for monitoring an ongoing job in a grid environment; the 'retrieve results' service is used to return the out of rendered job to the user in JPG format; and the 'encode result' service is used to convert rendered 3D images into a 3D animation or movie.

Further, this render grid solution has been tested with 400 frames of animation from a 3D model of Fishbourne Roman Villa Palace. This resulted in the rendering of the whole job in 55 minutes on six executer machines which turned out to be 85% faster than running it on a single machine of similar configuration. While this is only a small test, it is expected to scale up with more machines simply because Condor high throughput computing itself is a proven concept.

6.2 Real-time Digital Puppetry solution digital heritage

The second part of this work involves the use of these models (rendered from a content creation service) into presentation and interaction services for digital heritage

environments. Here the idea was to create digital puppets that can be driven in real-time to demonstrate their existence, and the existence of other heritage objects inside the digital heritage environments.

6.2.1 Motion capture based visualization using digital puppetry

The motion capture based online digital puppetry solution has been designed for digital heritage environments where a person wearing a motion capture suit (eMove suit) can drive a digital heritage avatar inside a digital heritage environment. The digital puppet can explain who he is, present his digital heritage objects to the audience, and show off his digital heritage environment, etc. The interactive sessions involves a dialogue between the digital puppet (character of the age) in real-time with the audience.

The current solution involves the use of an eMove motion capture suit and Autodesk MotionBuilder software for real-time 3D animations and interactions. Further enhanced work shows the possibility of using open source, freeware and low cost games engines to implement the same digital puppetry application at low cost.

6.2.2 Motion capture based interaction

The digital puppets and other digital heritage objects are controlled using a low cost upper body motion capture suit (eMove). This work has opened up new dimensions for digital heritage interaction techniques. These motion capture based interaction technologies connected with a game engine such as Unity3D or Panda3D; a 3D virtual heritage environment and objects modelled inside the gaming engine can be placed inside a museum (as a bespoke kiosk based interactive), or at an archaeological site (again as a kiosk) for visitors to ‘play’ with digital heritage objects. Also, this type of interaction can be used for creating heritage role-play, where different real life actors can play different roles from a particular heritage story.

This research work also includes the design and implementation of real-time motion capture drivers for Unity3D and Panda3D game engines to provide various low cost virtual 3D environments for exploitation by the low cost motion capture suit (eMove).

6.2.3 Audio and video technologies for digital heritage

A part of this research work includes the implementation of audio and video chat (online Internet based) capabilities (called the eMove-chat application); this enables a remote version of a digital puppetry solution, digital puppeteer to control a puppet from a remote location. This is particularly useful where the digital puppeteer and puppet are not at the same location. For example, a real life film actor can be hired to play a digital heritage puppet, which is installed inside many museums and archaeological sites; hence this actor can control all the avatars from his original location. He does not need to travel to site; this can surely save money for heritage show organizers. Clearly, there are consumer level applications that are possible too.

The motion capture based digital puppetry solution has been tested locally, in a local area network, a metropolitan area network, and even on a wide network. The successful test results conclude that this solution can easily be adopted for museum and archaeological sites (i.e. Intranet sites). Hence, adoption of real-time digital puppetry solutions will make these sites and museums more attractive for everyone.

6.3 Summary

The work carried out in this thesis is part of building a digital heritage system framework called DISPLAYS. This work was focused around rendering digital heritage contents, presenting them inside a digital heritage context, and interacting with these contents inside digital heritage environments using motion capture technologies. The first result describes the DISPLAYS concept. The second part describes the successful implementation of render grid, its applications and benefits for the heritage community by adding it as a part of a digital heritage system. The third part of this work was concerned with the implementation of digital puppetry for digital heritage for interaction and presentation scenarios. Both the render grid and the digital puppet solutions have been tested for performance, and their results appear at the end of their respective chapters.

6.4 Future work

The work presented inside this thesis, i.e. render grid and digital puppetry solutions are both unique in their scope, and these are especially unique for digital heritage systems. The implementation work carried out as part of this thesis was to prove the concepts and show that they actually work. More work needs to be done in order to make them publicly available, or indeed create products: this is of course beyond the scope of this thesis. However, in collaboration with Animazoo UK the digital puppetry solution is being created for entertainment and theme park scenarios.

Future work is identified below:

1. Render grid
    - The render grid solution needs to be packaged up as an installation file by integrating Condor grid core components inside the installation package. So when the user installs it, it should install everything at once; this will save the render grid user from worrying about the grid infrastructure and concentrate on his modeling.
    - The new installable solution then needs to be tested by providing it to an open source community such as Sourceforge.org or Blender.org, so that the community members can download and use it, and provide their feedback.
    - At present, the render grid solution is only developed for Blender 3D software. More research needs to be done to implement similar solutions for 3ds Max and Maya software.
2. Digital puppetry
    - The digital puppetry solution presented in this works depends on MotionBuilder, which is quite an expensive solution; there is still a need for developing a cost effective solution. Therefore, during the last year new real-time motion capture drivers have been developed for various game engines to produce a low cost version using freeware or open source game engines. In addition, more work is required to build a transformation process of
        - sound to phoneme, and
        - phoneme to visemes generation

all in real-time. For the first part (from sound to phoneme transformation) various SDKs have been tested (but this work is not part of this thesis) including Annosoft microphone, Papagayo, FacePoser, etc., and for phonemes to visemes transformation using Blender and Unity game engines. This work is currently ongoing.

- The models and environments used during these experiments were taken from Animazoo's existing theme park environments for proof of the concept. Future work involves designing new digital heritage environments, avatars, and objects, to play a real digital heritage puppet specifically for digital heritage scenarios.

7 *References*

[1] L. Candele and et al., *The DELOS Digital Library Reference Model. Foundations for Digital Libraries*. Glasgow: ISTI-CNR, December 2007.

[2] Leonardo Candela and et al., "DILIGENT: integrating digital library and Grid technologies for a new Earth observation research infrastructure," vol. 7 / October, 2007, no. Numbers 1-2, October, 2007.

[3] Predrag Knezevic, Carlo Meghini, Carlo Meghini, and Fiore Basile Thomas Risse, "The BRICKS Infrastructure - An Overview," , Moscow, 2005.

[4] Martin White, et al "ARCO—An Architecture for Digitization, Management and Presentation of Virtual Exhibitions," in *Proceedings of the Computer Graphics International*, 2004.

[5] Beyond the Text Project - Reanimating cultural heritage: digital repatriation, knowledge networks and civil society strengthening in post-conflict Sierra Leone. (2010, July) [Online]. http://projects.beyondtext.ac.uk/reanimatingculturalheritage/index.php

[6] A.Al-Barakati, M.Gkion, W.Zhang, P. Newbury, N. Beloff, and M.White M.Z.Patoli, "A Service-Orientation Approach for a Digital Library System focused on Portable Antiquities and Shared Heritage," , Brighton, 2007.

[7] M.Z.Patoli, M.Gkion, W.Zhang, P. Newbury, N. Beloff, and M.White A.Al-Barakati, "A Dynamic Workflow Management Framework for Digital Heritage and Technology Enhanced Learning," , ARCAEOLINGUA, Cyprus, 978-963-9911-00-0, 2008.

[8] Written Communication vs. Visual Communication, Gaiam Life. (2010, Mar.) [Online]. http://blog.gaiam.com/quotes/topics/insight

[9] Michael Bell Eric A. Marks, "Conceptual SOA Vision," in *Executive's Guide to Service-Oriented Architecture*. USA: John Wiley & Sons, Inc., 2006.

[10] New to SOA and Web services, IBM. (2010, Mar.) [Online]. http://www.ibm.com/developerworks/webservices/newto/

[11] Daniel Minoli, *A Networking approach to Grid computing*. Canada: John Wiley & Sons, 2005.

[12] Ian Foster and Carl Kesselman, *The Grid : Blueprint For A New Computing Infrastructure*.: Morgan Kaufmann, 1999.

[13] Ian Foster and Carl Kesselman, "Computational Grid (Invited Talk) (Page-3 to 37)," 2001.

[14] Ian Foster, "What is the Grid? A Three Point Checklist," Chicago, July 20, 2002.

[15] Ian Foster and Carl Kesselman, *The Grid 2 Blueprint for a New Computing Infrastructure*. San Francisco: Elsvier Inc, 2004.

[16] Hai Zhuge, *The knowledge grid*.: World Scientific, 2004.

[17] April J. Wells, *Grid Database Design*.: Auerbach Publications; 1 edition, ISBN: 978-0849328008, 26 May 2005.

[18] Domenico Talia Mario Cannataro, "The Knowledge Grid," 2003.

[19] Welcome to KnowledgeGRID Malaysia. (2009, June) [Online]. https://www.knowledgegrid.net.my/index.jsp?p=

[20] Johan Tordsson Erik Elmroth, "A Grid Resource Broker Supporting Advance Reservations and Benchmark-Based Resource Selection ," vol. 3732, February 27, 2006.

[21] David Abramson, and Jonathan Giddy Rajkumar Buyya, "An Economy Driven Resource Management Architecture for Global Computational Power Grids," , Las Vegas, 2000.

[22] David Abramson, Jonathan Giddy, and Heinz Stockinger Rajkumar Buyya, "Economic models for resource management and scheduling in Grid computing," vol. 14, 2002.

[23] Grid computing - distributed computing, Maxi Pedia. (2010, Jan.) [Online]. http://www.maxi-pedia.com/Grid+computing+distributed+computing

[24] D. Britton and et al, "GridPP: Meeting the Particle Physics Computing Challenge," Proc. Of UK e-Science All Hands Meeting, 2005.

[25] Osamu Tatebe and et al, "Grid Datafarm Architecture for Petascale Data Intensive Computing," , 2002.

[26] Chaitanya Baru, Reagan Moore, Arcot Rajasekar, and Michael Wan, "The SDSC Storage Resource Broker," , Proceedings of the 1998 conference of the Centre for Advanced Studies on Collaborative research, Toronto, Canada, 1998.

[27] Chao-Tung Yang and et al, "A Network Bandwidth-Aware Job Scheduling with Dynamic Information Model for Grid Resource Brokers," , Asia-Pacific Services Computing Conference, APSCC '08. IEEE, ISBN: 978-0-7695-3473-2, 2008.

[28] Karl Czajkowski et al, "Resource Co-Allocation in Computational Grids," , California , 2006.

[29] The Gridbus Middleware, Cloudbus. (2010, Feb.) [Online]. http://www.cloudbus.org/middleware/

[30] Rajkumar Buyya, Rajiv Ranjan, Srikumar Venugopal Akshay Luther, "Alchemi: A.NET-based Grid Computing Framework and its Integration into Global Grids," University of Melbourne, Melbourne, Australia, GRIDS-TR-2003-8, 2003.

[31] Rajkumar Buyya, Rajiv Ranjan, and Srikumar Venugopal Akshay Luther, "Alchemi: A.NET-based Enterprise Grid Computing System," , Proceedings of the 6th International Conference on Internet Computing (ICOMP'05), Las Vegas, USA, 27-30, 2005.

[32] Rajkumar Buyya, "The Gridbus Toolkit for Grid and Utility Computing," , Fifth IEEE International Conference on Cluster Computing (CLUSTER'03), ISBN: 0-7695-2066-9, Hong Kong, 2004.

[33] Manzur Murshed Rajkumar Buyya, "GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing," vol. 14, no. 13-15, 2003.

[34] Rajkumar Buyya, Alexander Barmouta, "GridBank: A Grid Accounting Services Architecture (GASA) for Distributed Systems Sharing and Integration," , International Parallel and Distributed Processing Symposium (IPDPS'03), Nice, France, 22-26 April, 2003.

[35] Todd Tannenbaum, and Miron Livny Douglas Thain, "Condor and the Grid," in *Grid computing: making the global infrastructure a reality*.: John Wiley & Sons, Ltd, 2003.

[36] David Abramson, Jonathan Giddy Rajkumar Buyya, "Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid," , 2000.

[37] Ian Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems," vol. 21, no. 4, July 2006.

[38] Craig Fellenstein Joshy Joseph, "Condor and Condor-G (Page 34)," in *Grid computing*. NJ United States: Pearson Education, Inc. (Prentice Hall), 2004.

[39] gLite Lightweight Middle ware for Grid, CERN. (2010, Mar.) [Online]. http://glite.web.cern.ch/glite/

[40] Globus Alliance. (2010, Mar.) [Online]. http://www.globus.org/

[41] GridPP UK Computing for Particle Physics. (2010, Feb.) [Online]. http://www.gridpp.ac.uk/explain.html

[42] About NGS, National Grid Services. (2010, Mar.) [Online]. http://www.ngs.ac.uk/aboutUs

[43] TeraGrid Organization. (2010, Mar.) [Online]. https://www.teragrid.org/web/about/

[44] David F. Snelling Dietmar W. Erwin, "UNICORE: A Grid Computing Environment," in *Lecture Notes in Computer Science*.: Springer Berlin / Heidelberg, 2001.

[45] Wolfgang Gentzsch and Palo Alto, "Sun Grid Engine: Towards Creating a Compute Power Grid," , IEEE Computer Society, Proceedings of the 1st International Symposium on Cluster Computing and the Grid (CCGRID '01), ISBN: 0-7695-1010-8, 2001.

[46] About OGSA-DAI. (2010, Mar.) [Online]. http://www.ogsadai.org.uk/about/index.php

[47] Tamas Kiss, Peter Kacsuk, Gabor Terstyanszky Tamas Kukla, "Integrating Open Grid Services Architecture Data Access and Integration with computational Grid workflows," vol. 367, June 2009.

[48] SETI @ Home, Berkeley University of California. (2010, Apr.) [Online]. http://setiathome.berkeley.edu/

[49] Climateprediction Home page. (2010, Apr.) [Online]. http://climateprediction.net/

[50] Baker Lab, Rosetta@home. (2010, Apr.) [Online]. http://boinc.bakerlab.org/rosetta/

[51] World Community Grid Project. (2010, Apr.) [Online]. http://www.worldcommunitygrid.org/

[52] BOINC, Berkeley. (2010, Mar.) [Online]. http://boinc.berkeley.edu/download.php

[53] Xgrid: High Performance Computing for the Rest of Us, Apple Developer Connection. (2010, Mar.) [Online]. http://developer.apple.com/hardwaredrivers/hpc/xgrid_intro.html

[54] GDI|Queue, 2001 - 2004. (2010, Jan.) [Online]. http://www.cirquedigital.com/products/gdiqueue/index.html

[55] Jennifer M. Schopf, "A General Architecture for Scheduling on the Grid," *Special Issue on Grid Computing, J. Parallel and Distributed Computing*, 2002.

[56] Ma Jie and Li Bo Ni Guangbao, "GridView: A Dynamic and Visual Grid Monitoring System," , Tokyo, Japan, 2004.

[57] GRAM, The Globus Alliance. (2010, Mar.) [Online]. http://dev.globus.org/wiki/GRAM

[58] et al. P. Kacsuk, "Demonstration of P-GRADE Job-Mode for the Grid ," vol. 2790, June 2004. [Online]. http://portal.p-grade.hu/

[59] portal.ahm.honda.com, Robtex.com. (2010, Mar.) [Online]. http://www.robtex.com/dns/portal.ahm.honda.com.html#summary

[60] P-Grade Protal Installations. (2010, Feb.) [Online]. http://portal.p-grade.hu/?m=installations&s=0

[61] Grid Portal Development Kit (GPDK), DOE Science Grid. (2010, Apr.) [Online]. http://doesciencegrid.org/projects/GPDK/

[62] EGEE (Enabling Grid for E-scienceE). (2010, Mar.) [Online]. http://www.eu-egee.org/

[63] GridWay. (2010, Mar.) [Online]. http://www.gridway.org/doku.php?id=start

[64] healthGrid.org. (2010, Apr.) [Online]. https://trac.healthgrid.org/grid-workshop/wiki/Ganga

[65] Peter H Beckman, "Building the TeraGrid," vol. 363, Phil. Trans. R. Soc. A, London, UK, ISBN: 1833 1715-1728, 2005.

[66] WLCG (Worldwide LHC Computing Grid), CERN. (2010, Mar.) [Online]. http://lcg.web.cern.ch/LCG/

[67] LHC Grid Fest, CERN. (2010, Mar.) [Online]. http://lcg.web.cern.ch/LCG/lhcgridfest/

[68] BRIDGES: Biomedical Research Informatics Delivered by Grid Enabled Services, University of Glasgow. (2010, Feb.) [Online]. http://www.brc.dcs.gla.ac.uk/projects/bridges/

[69] Harvesting Spare CPU cycles - The Monash Green SPONGE, Monash University. (2010, Mar.) [Online]. http://www.monash.edu.au/eresearch/activities/sponge.html

[70] Avatar, Box Office Mojo. (2010, June) [Online]. http://www.boxofficemojo.com/movies/?id=avatar.htm

[71] List of highest-grossing films, WikiPedia. (2010, June) [Online]. http://en.wikipedia.org/wiki/List_of_highest-grossing_films#cite_note-1

[72] The Art of Motion Capture in Avatar, Vizworld. (2010, June) [Online]. http://www.vizworld.com/2009/12/art-motion-capture-avatar/#more-10487

[73] Matt Liverman, *The Animator's Motion Capture Guide Organizing, Managing, and Editing*.: Charles River Media, 2004.

[74] Alberto Menache, "Motion Capture Primer," in *Understanding motion capture for computer animation and video games*. San Diego, USA: Academic Press, 1995.

[75] Autodesk MotionBuilder, Autodesk. (2010, Apr.) [Online]. http://usa.autodesk.com/adsk/servlet/pc/index?siteID=123112&id=13581855

[76] Melissa Gross Elizabeth Crane, "Motion Capture and Emotion: Affect Detection in Whole Body Movement," vol. 4738, 2007.

[77] Brian Windsor Midori Kitagawa, "Types of Mocap," in *MoCap for Artists: Workflow and Techniques for Motion Capture*.: Focal Press, 2008.

[78] Jeff Ridenour, David J. Cuccia Frédéric Bevilacqua, "3D motion capture data: motion analysis and mapping to music," , Santa Barbara, California, USA , 2002.

[79] Vicon motion capture systems. (2010, Feb.) [Online]. http://www.vicon.com/index.html

[80] Motion Analysis - Home page (2010, Mar.). [Online]. http://www.motionanalysis.com/

[81] Wikipedia (Motion Capture). (2010, Aug.) [Online]. http://en.wikipedia.org/wiki/Motion_capture

[82] Phasespace home page. (2010, Feb.) [Online]. http://www.phasespace.com/index.html

[83] Optotrak Certus Motion Capture System, NDIgital. (2010, Mar.) [Online]. http://www.ndigital.com/lifesciences/certus-motioncapturesystem.php

[84] INITION - PhaseSpace. (2010, July) [Online]. http://www.inition.co.uk/inition/dispatcher.php?action=get&model=products&URL_=product_mocaptrack_phasespace&SubCatID_=59&tab=blurb

[85] VANDERBILT - Motion Capture Example. (2010, July) [Online]. http://www.vuse.vanderbilt.edu/~bobbyb/images/Race_bolts.html

[86] Animazoo UK, LTD, Animazoo.com. (2010, Mar.) [Online]. http://www.animazoo.com/

[87] Nadia Bianchi-Berthouze, Betsy van Dijk, and Anton Nijholt Marco Pasch, "Movement-based sports video games: Investigating motivation and gaming experience ," vol. 1, no. 2, April 2009, Pages 49-61.

[88] C Père, N Meylaender, and F Mérienne, "Full body motion capture in CAD environment," , Proceedings of Virtual Concept 2006, Cancún, Maxico, 2006.

[89] Gypsy6 / Gypsy6 Torso, INITION. (2010, July) [Online]. http://www.inition.co.uk/inition/dispatcher.php?action=get&model=products&URL_=product_mocaptrack_animazoo_gypsy&SubCatID_=0&tab=blurb

[90] Martin White and et al, "4D TRAVEL (culTural heRitAge Visual experiences through motion capturE and hoLoscopic displays)," Brighton (contact M.White@sussex.ac.uk for full copy of the report), 2010.

[91] Raphaël Sebbe, Thierry Dutoit Joëlle Tilmanne, "A DATABASE FOR STYLISTIC HUMAN GAIT MODELING AND SYNTHESIS," vol. 1, no. 3, September 2008, pp. 97-100.

[92] IGS-190-M, Animazoo.com. (2010, May) [Online]. http://www.animazoo.com/index.php/igs-190-hybrid

[93] Xsens MVN - Inertial Motion Capture. (2010, July) [Online]. http://www.xsens.com/en/general/mvn

[94] Project Natal, XBox.com. (2010 , Apr.) [Online]. http://www.xbox.com/en-us/live/projectnatal/

[95] Blender Foundation. (2010, Mar.) Blender 3D, ender.org. [Online]. http://www.blender.org/

[96] About Magpie Pro, Third Wish Software and Animation. (2010, Mar.) [Online]. http://www.thirdwishsoftware.com/magpiepro.html

[97] iClone 4, Reallusion. (2010, May) [Online]. http://www.reallusion.com/iclone/

[98] Crazy Talk, Reallusion. (2010, May) [Online]. http://www.reallusion.com/crazytalk/

[99] Papagayo, Digital Arena. (2010, Mar.) [Online]. http://www.digitalarena.co.uk/animation/papagayo.htm

[100] Source Engine, Valve Business Solutions. (2010, Mar.) [Online]. http://source.valvesoftware.com/

[101] Facebook. (2010, Jan.) [Online]. http://www.facebook.com/

[102] Annosoft, LLC. (2010, Mar.) [Online]. http://www.annosoft.com/

[103] 3D Game Engines Database, DevMaster.net. (2010, June) [Online]. http://www.devmaster.net/engines/

[104] Panda3d.org. (2009, Dec.) [Online]. http://www.panda3d.org/

[105] Unity 3D home Page. (2009, Dec.) [Online]. http://unity3d.com/

[106] Unreal Tournament, EPIC Games. (2010, Apr.) [Online]. http://www.epicgames.com/

[107] Unreal Engine, Wikipedia. (2010, Apr.) [Online]. http://en.wikipedia.org/wiki/Unreal_Engine

[108] DAVID BEARMAN, "ADVANCED QUIXIS," *Archives and Museum Informatics*, vol. 4, no. 2, SUMMER 1990.

[109] About Squiz: The Supported Open Source CMS Solutions Company, Squize. (2010, Aug.) [Online]. http://www.squiz.co.uk/about

[110] Check Out Open Source Library and Collections Management Software, Linux.com. (2010, Aug.) [Online]. http://www.linux.com/news/enterprise/case-studies/26661-check-out-open-source-library-and-collections-management-software

[111] M. Z. Patoli, M. Gkion, Al-Barakati, P. Newbury and M. White W.Zhang, "Reanimating Cultural Heritage through Service Orientation, Workflows, Social Networking and Mashups," , Bradford, UK, 2009.

[112] Rafa Wojciechowski, Krzysztof Walczak, Martin White, and Wojciech Cellary, "Building Virtual and Augmented Reality Museum Exhibitions," in *Proceedings of the ninth international conference on 3D Web technology, ACM*, Monterey, California, 2004.

[113] Zeeshan Patoli et al., "How to Build an Open Source Render Farm based on Desktop Grid Computing," , Springer CCIS, Karachi, ISBN: 978-3-540-89852-8, 2008.

[114] M. Gkion, A. Al-Barakati, W. Zhang, P. Newbury and M. White M. Z. Patoli, "An Open Source Grid Based Render Farm for Blender 3D ," , Seattle, USA, 2009.

[115] BRICKS - Building Resources for Integrated Cultural Knowledge Services, European Information and Communication Technologies. (2010, Aug.) [Online]. http://cordis.europa.eu/ist/digicult/bricks.htm

[116] Krzysztof Walczak, Fabrizio Giorgini, Martin White Manjula Patel, "A Cultural Heritage Repository as Source for Learning Materials VAST (2004)," in *The 5th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (The Eurographics Association 2004)*, Brussels, 2004.

[117] W. Zhang, M. Z. Patoli, M. Gkion, P. Newbury and M. White A. Al-Barakati, "An Integrated Workflow Management Solution for Heritage Information Mashups," , IEEE Computer Society, Athens, Greece, 978-0-7695-3689-7, 2009.

[118] M. Z. Patoli, Al-Barakati, W. Zhang, P. Newbury and M. White M. Gkion, "Collaborative 3D Digital Content Creation Exploiting a Grid Network," , Karachi, 2009.

[119] M. Gkion, P. Newbury and M. White M. Z. Patoli, "Real Time Online Motion Capture for Entertainment Applications," , Kaohsiung, Taiwan, 2010.

[120] M. Gkion and M. White M. Z. Patoli, "Real-time Online Digital Avatar with Lip Syncing and Facial Expression," in *ECRT (Early Carrier Researcher's Track) Proceeding of 3rd IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL 2010), ISBN: 978-986-02-2832-8.*, Feb 2010.

[121] eMove Team. (2008, December) www.sussex.ac.uk. [Online]. http://www.informatics.sussex.ac.uk/research/groups/cvcg/emove/

[122] UnReal Home Page. (2010, Jan.) [Online]. http://www.unreal.com/

[123] Microsoft. XAML, MSDN.Net Framework Developer Center. (2009, Dec.) [Online]. http://msdn.microsoft.com/en-us/library/ms752059.aspx

[124] COLLADA - Digital Asset and FX Exchange Schema. (2009, Nov.) [Online]. https://collada.org/mediawiki/index.php/COLLADA_-_Digital_Asset_and_FX_Exchange_Schema

[125] NeuroVR Home Page. (2010, Mar.) [Online]. http://www.neurovr.org/index.php?option=com_frontpage&Itemid=1

[126] ARC 3D Webservice, ESAT. (2009, Dec.) [Online]. http://homes.esat.kuleuven.be/~visit3d/webservice/v2/index.php

[127] Martin White, Krzysztof Walczak, Patrick Sayd Manjula Patel, "Digitisation to Presentation –

Building Virtual Museum Exhibitions," in *Proceedings of International Conference on Vision, Video and Graphics*, Bath, UK, July 2003.

[128] Apache Tomcat, Apache. (2009, Dec.) [Online]. http://tomcat.apache.org/

[129] LOCKSS, stanford. (2010, Jan.) [Online]. http://lockss.stanford.edu/lockss/Home

[130] Mashups, Wikipedia. (2010, Jan.) [Online]. http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid)

[131] Alexander Mikroyannidis, "Toward a Social Semantic Web," , Los Alamitos, CA, USA, 2007.

[132] Virtex. Interactive Institute. (2009, May) [Online]. www.tii.se/knowhow

[133] Santa Chiara Church, Victoria and Albert Museum. (2010, May) [Online]. http://www.arco-web.org/Virtual/santachiarainstallation.php

[134] VRML, Wikipedia. (2010, Jan.) [Online]. http://en.wikipedia.org/wiki/VRML

[135] Cortona3D viewer, Cortona3D. (2010, Jan.) [Online]. http://www.cortona3d.com/Products/Cortona-3D-Viewer.aspx

[136] IBM Digital Media: Solution for the Media and Entertainment Industry. (2005, May) Building an animation and special effects studio from the ground up. http://www-304.ibm.com/businesscenter/cpe/download6/37646/Animation_WP_060105.pdf. [Online]. Building an animation and special effects studio from the ground up

[137] Condor High Throughput Computing, Computer Science department, University of Wisconsin Madison. (2010, Mar.) [Online]. http://parrot.cs.wisc.edu/

[138] Clovis Chapman and al et, "Condor Birdbath: Web Service interface to condor," in *UK e-Science All Hands Meeting, ISBN 1-904425-53-4*, Nottingham, UK, September 2007, pp. 737-744.

[139] R. Andersen, "Harvesting Idle Windows CPU Cycles for Grid Computing," , Proceedings of the 2006 International Conference on Grid Computing & Applications, GCA 2006, Las Vegas, Nevada, USA, June 26-29, 2006.

[140] DrQueue Background. (2010, Mar.) [Online]. http://www.drqueue.org/cwebsite/about_drqueue.php

[141] Lobo's Blender Page, FarmerJoe. (2010, Mar.) [Online]. http://blender.formworks.co.nz/?p=1

[142] Gordon Stoll, Homan Igehy, Kekoa Proudfoot and Pat Hanrahan Milton Chen, "Models of the Impact of Overlap in Bucket Rendering," , 1998.

[143] Gary Bishop Leonard McMillan, "Plenoptic Modeling: An Image-Based Rendering System," , 1995.

[144] Loki 0.6.2, Loki Render. (2010, Feb.) [Online]. http://loki-render.berlios.de/

[145] burp | renderfarming.net - The Big and Ugly Rendering Project, Renderfarming. (2010, Apr.) [Online]. http://burp.renderfarming.net/

[146] About Deadline, Emecstudios. (2010, Mar.) [Online]. http://deadline.emecstudios.com/index.php/about.html

[147] You need Respower. (2010, Mar.) [Online]. http://www.respower.com/

[148] RenderCore. (2010, Mar.) [Online]. http://www.rendercore.com/

[149] "Converting video formats with FFmpeg," no. 146, 2006.

[150] RealVNC, © RealVNC Limited. (2010, May) [Online]. http://www.realvnc.com/

[151] Radmin Remote Control Software. (2010, May) [Online]. http://www.radmin.com/

[152] Teamviewer. (2010, May) [Online]. http://www.teamviewer.com/index.aspx

[153] About SQLite Database Engine, SQLite.org. (2010, May) [Online]. http://www.sqlite.org/about.html

[154] "Real-Time Photo Realistic Simulation of Complex Heritage Edifices," , Berkeley, California, 25th October 2001, pp. IEEE Computer society, Seventh International Conference on Virtual Systems and Multimedia (VSMM'01).

[155] DTU Electrical Engineering, Technical University of Denmark. (2010, June) [Online]. http://server.elektro.dtu.dk/personal/oldat/erato/

[156] Naif Haddad, "Criteria for the Assessment of the Modern Use of Ancient Theatres and Odea," vol. 13, no. 3, 2007.

[157] CHARACTERISE - Virtual Human Open Simulation Framework for Cultural Heritage, EPOCH European Network of Excellence in Open Cultural Heritage. (2010, June) [Online]. http://www.epoch.eu/index.php?option=com_content&task=view&id=227&Itemid=343

[158] David Arnold, et al. "Tools for Populating Cultural Heritage Environments with Interactive Virtual Humans," , Brighton, 2008.

[159] Daniel Pletinckx , Katerina Mania , Martin White Panagiotis Petridis, "The EPOCH Multimodal Interface for Interacting with Digital Heritage Artefacts," , 2006.

[160] Katerina Mania, Daniel Pletinckx, Martin White Panagiotis Petridis, "Usability evaluation of the EPOCH multimodal user interface: designing 3D tangible interactions," , Limassol, Cyprus , 2006.

[161] LEE HARRISON III, ANIMAC (Hybrid graphic animation computer), 1962, vasulka.org. (2010, May) [Online]. http://www.vasulka.org/Kitchen/PDF_Eigenwelt/pdf/092-095.pdf

[162] David J. Sturman. Siggraph. [Online]. http://www.siggraph.org/education/materials/HyperGraph/animation/character_animation/motion_capture/history1.htm

[163] Centre for VLSI and Computer Graphics, University of Sussex. (2010, June) [Online]. http://www.informatics.sussex.ac.uk/research/groups/cvcg/

[164] Technology Strategy Board UK, Innovateuk.org. (2010, Mar.) [Online]. http://www.innovateuk.org/

[165] B. Goode, "Voice over Internet protocol (VoIP)," 2002.

[166] Mauro Tropea, Peppino Fazio, Salvatore Marano Floriano De Rango, "Overview on VoIP: Subjective and Objective Measurement Methods," *IJCSNS International Jour 140 nal of Computer Science and Network Security*, vol. 6, no. 1B, pp. 140 - 153, January 2006.

[167] Yoshikazu Yokotani and Soontorn Oraintara, "LOSSLESS AUDIO COMPRESSION USING INTEGER MODIFIED DISCRETE COSINE TRANSFORM," in *2003 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2003)*, Awaji Island, Japan, December 2003, pp. 120 - 126.

[168] Xie Xiao-Gang, Cai Jun, Chen Qi-Chuan, and Ou Jian-Lin, "Design and implementation of VoIP system based on Speex codec," vol. 24, no. 12, Application Research of Computers, 2007.

[169] W.B. Kleijn, R. Hagen, J. Linden, M.N. Murthi, J. Skoglund S.V. Andersen, "iLBC - a linear predictive coder with robustness to packet losses," in *Speech Coding, 2002, IEEE Workshop Proceedings, ISBN: 0-7803-7549-1* , Oct 2002, pp. 23 - 25.

[170] DU Baojiang, XIONG Xiantao WANG Yequan, "Real-time Video Capture of the Virtual Studio," 2008.

[171] Howard A. Seid and Albert Lespagnol, "Virtual private network," 5,768,271, Jun 16, 1998.

[172] M. Feilner, *OpenVPN: Building and Integrating Virtual Private Networks*.: safari Books online, 2006.

[173] D.J. Morrow, T. Littler D. Laverty, "INTERNET BASED LOSS-OF-MAINS DETECTION FOR DISTRIBUTED GENERATION," in *Universities Power Engineering Conference, 2007. UPEC 2007. 42nd International* , Brighton, Sept. 2007, pp. 464 - 469.

Appendix – A

Use Case for Digital Puppetry

Initialize Client Side

Start ‘online animation, audio and video chat system’

Use Case #	UC001
Use case name	Initialize Client Side Start the ‘online animation, audio and video chat system’ (client side)
Description	A system administrator starts ‘online animation, audio and video chat system’ on the client. This involves starting client’s webcam so that the person wearing a eMove suit can see who is coming in front of screen, connecting voice chat and setting up MotionBuilder, to connect 3D avatar with local or remote mocap data. • Start AnimaDemo software and start audio and video chat. • Setup MotionBuilder (i.e. 3D character, network settings) so that animation and voice data can be fed by eMove server.
Scope and level	Heritage museums, sites, theme parks, and heritage theatre shows
Primary and secondary actors	System Administrator
Preconditions	1. Hardware and software should already be installed 2. Animation data should be served from server
Stakeholders and interests	Archaeologists, Heritage lovers, common people
Triggers	A System administrator starts eMove client by initializing eMove-chat client and virtual environment inside MotionBuilder.
Primary scenario	1. Start Hamachi 2. Connect MotionBuilder to eMove server 3. Start eMove-chat 4. Start eMove-chat webcam
Success guarantee (Post conditions)	eMove client including eMove-chat client and MotionBuilder should be running and connected to eMove server
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 17/03/09

Initialize Server Side

Use Case #	UC002
Use case name	Initialize Server Side Start the 'online animation, audio and video chat system' (server side)
Description	A system administrator starts 'online animation, audio and video chat system' on the server. This involves starting initializing eMove-chat client, AnimaDemo and streaming the mocap data to eMove-client application (i.e. MotionBuilder) • Start Hamachi • Start AnimaDemo software and start audio and video chat. • Stream mocap data to the client • Starting eMove-chat server software • Start client's video
Scope and level	Heritage museums, sites, theme parks, and heritage theatre shows
Primary and secondary actors	A person wearing eMove suit will interact with a person as a 3D avatar inside heritage environment (e.g. museum or heritage site)
Preconditions	1. Hardware and software should already be installed 2. eMove suit should be connected to the system and worn by eMove actor
Stakeholders and interests	Child, theme park, …
Triggers	eMove actor starts Anima Server components
Primary scenario	1. Starts Hamachi 2. Start Anima Chat 3. Start Anima Video
Success guarantee (Post conditions)	Anima Server including Anima Chat Server and Anima Suit data should be running and ready to be connected by client. eMove Actor can see who is coming inside the both through Anima Webcam
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 17/03/09

Select 3D Avatar

Use Case #	UC003
Use case name	Select 3D Avatar
Description	Child enters the Online animation, audio and video chat system booth and selects the desired 3D animated character. Alternatively, a default animated 3D Avatar is already selected for efficiency, perhaps because the Online animation, audio and video chat system is specifically set up for a user context, e.g. already set up for Father Christmas.
Scope and level	Scenario 1, elements may be applicable to scenario 2.
Primary and secondary actors	The child.
Preconditions	System should be turn on and ready to go.
Stakeholders and interests	Child, theme park, …
Triggers	Child starts the Online animation, audio and video chat system by selection the 3D animated character, or if already started the 3D animated character starts to talk to the child.
Primary scenario	1. Child enters the booth 2. Child selects character 3. Child
Alternative scenario	1. Child starts talking to default 3D animated character, e.g. Father Christmas
Success guarantee (Post conditions)	Child successfully interacts with the 3D animated character
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 16/03/09

Call to 3D Avatar

Use Case #	UC004
Use case name	Call to 3D Avatar
Description	A child calls to 3D Avatar to interact with the person wearing eMove suite, The eMove actor accepts the call and starts interacting with the child. This use case involves connections for Audio, text, and Animation data. Alternatively the Connection should be established on selecting 3D Avatar from list of options.
Scope and level	Scenario 1
Primary and secondary actors	Primary Actor : Child Secondary Actor: A person wearing eMove suit
Preconditions	Child should select a 3D avatar or the default one should be selected
Stakeholders and interests	Child, theme park, …
Triggers	A Child after selecting 3D Avatar calls him for interaction. Alternatively it should be done on selection of 3D avatar
Primary scenario	1. Child press call button
Alternative scenario	Alternatively this should all be done on selection of 3D Avatar
Success guarantee (Post conditions)	A child in booth and actor wearing eMove suit should be able to talk and interact with each other.
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 17/03/09

Audio Video Text Chat

Use Case #	UC005
Use case name	Audio Video Text Chat
Description	During the Audio, video and Animation interaction, sometimes, if child or eMove actor cannot understand pronunciation or a word then can type in Anima Text Chat facility and send it to other party
Scope and level	Scenario 1, elements may be applicable to scenario 2.
Primary and secondary actors	Child and eMove Actor
Preconditions	Anima Chat Client should be connected
Stakeholders and interests	'online animation, audio and video chat system', Theme Park, etc...
Triggers	Customer / eMove actor type words or sentences and press send button to send it to other party
Primary scenario	eMove Actor and child interact with each other using text chat facility
Alternative scenario	Both actor and child continues to use Audio chat and have no difficulty to understand each other, hence they don't use text chat facility.
Success guarantee (Post conditions)	Both eMove Actor and customer interacts with each other
Notes	See scenario description. Section 1.5.2.
Author and date	Zeeshan Patoli 17/03/09

Change 3D Avatar

Use Case #	UC006
Use case name	Change 3D Avatar
Description	A Child can change 3D avatar during his ongoing chat with existing Avatar. The eMove actor will be notified by the system so that he can respond accordingly
Scope and level	Scenario 1
Primary and secondary actors	Child / Customer
Preconditions	Anima Client components should be connected with Anima server components.
Stakeholders and interests	'Online animation, audio and video chat system', Theme Park etc...
Triggers	Child / Customer click on change avatar option to chose a difference 3D avatar from the list of available 3D Avatars
Primary scenario	During a chat session a child may want to talk to another avatar instead of currently selected Avatar
Alternative scenario	1. Child continues to talk existing avatar for whole of his chat session 2. Only one 3D Avatar is available
Success guarantee (Post conditions)	A child in a booth has successfully replaced new 3D Avatar with old one.
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 17/03/09

Joystick Actions

Use Case #	UC007
Use case name	Joystick Actions
Description	An eMove actor may want to perform different body and facial expressions using Joystick keys built-in into the eMove suit
Scope and level	Scenario 1 and 2
Primary and secondary actors	eMove Actor
Preconditions	Both eMove client and server should be connected and chat session should have been established. eMove suit with Joysticks must be installed to allow Joystick actions.
Stakeholders and interests	'online animation, audio and video chat system', Theme Park, etc...
Triggers	eMove actor can use different joystick keys for different actions / events
Primary scenario	An eMove actor may want to use joystick to perform different actions like • Resetting 3D Avatar position • Facing to the North (for calibration activity) • Some facial expressions • Some addition body movements Or other similar actions
Success guarantee (Post conditions)	eMove actor perform different body movements and facial expressions using built-in joystick in eMove suit.
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 17/03/09

Voice Pitch

Use Case #	UC008
Use case name	Voice Pitch
Description	Anima actor can sometimes change his voice pitch to present truly as a 3D avatar, e.g. Father Christmas or any other avatar. This will reduce the age difference between 3D Avatar and actual eMove actor
Scope and level	Scenario 1 and 2
Primary and secondary actors	eMove Actor
Preconditions	The Anima chat client server system should be up and running
Stakeholders and interests	'online animation, audio and video chat system' and Theme Park
Triggers	Can Change different voice pitch
Primary scenario	A person wearing eMove suit feels his voice pitch does not meets the customer's selected 3D Avatar, and he decides to change his voice pitch from Anima Chat to act more closely to selected 3D avatar.
Alternative scenario	• eMove actor does not change the voice pitch and decides to continue with the same voice • There is constant voice pitch for all actor • Or there is only one actor in the system.
Success guarantee (Post conditions)	A person wearing eMove suit successfully selected a new voice pitch that matches 3D Avatar and meets the requirement of originality
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 17/03/09

3D Animation

Use Case #	UC009
Use case name	3D Animation
Description	This use case describes the animation flow from AnimaServer to AnimaClient.
Scope and level	Scenario 1, 2, 3
Primary and secondary actors	eMove Actor
Preconditions	Anima Suit should be connected to AnimaServer The Client should be connected to AnimaServer A 3D avatar should be selected
Stakeholders and interests	'online animation, audio and video chat system' , Theme Park, and all other applications
Triggers	Animation data from server to client
Primary scenario	When user / child selects 3D character, the eMove actor receives a dialogue of notification about the character. The eMove actor then selects starts serving the animation data that is feed to 3D avatar inside MotionBuilder and the user / child can see a moving and animating 3D avatar
Alternative scenario	N/A
Success guarantee (Post conditions)	The user / child will see an animating 3D avatar on screen
Notes	See scenario description. Section 1.5.2
Author and date	Zeeshan Patoli 17/03/09

Appendix – B

Setting up Render Grid Environment

Desktop Grid Computing Environment Render Farm Setup

The setup of the desktop Grid based 'render grid' solution required three main components: a **Central Manager**, a job **Submitter** and multiple **Executers** or servers. To test the Condor Grid middleware it was possible to install all components on a single machine. But for an actual render farm solution, more than one executer could be used. For the first test, six old machines were used as described above for the **Executers**. The functionality of each component can be described as:

1. The **Central Manager** of the desktop Grid is a machine that holds the scheduler for the Grid that will distribute the render jobs to different executers (sometimes called servers). To operate Condor, the user needs to install the Negotiator, Collector and Master components on the machine acting as the central manager.
2. The job **Submitter** is the machine that can submit the render jobs to desktop Grid. To operate Condor correctly a job submitter machine must have the Scheduler and Master components installed.
3. The **Executer** machines each act as a server that gets the job from the Central Manager and executes them. These machines must have Start and Master components to provide the services. These machines must also have the application component installed, e.g. the Blender rendering component.

The Condor Software Installation

The installation of Condor on the Windows operating system is not as difficult as other Grid tools, which is another reason for choosing Condor. However, we also tried out Globus (Linux based) and the 3ds Max Backburner solution mentioned above. Condor can be downloaded free of cost with its instruction manual, etc. from University of Wisconsin website for different operating systems [8]. The Condor has nice user-friendly installation wizard which takes no more than five minutes to install on single machine and then some changes must be made in the Condor_config file for proper configuration. it is important that the the following changes are applied in the Condor_config file after installing it on Windows:

1. Select TESTINGMODE instead of UWCS mode set in default.

2. Collector name should be the name of central manager.
3. Hostallow_Read should be set to all names of machines which can view the status of a computer pool.
4. Hostallow_Write should be set to all names of machines which can join the pool and can submit the job to it.
5. Comment out the fields NEGOTIATOR_PRE_JOB_RANK. NEGOTIATOR_POST_JOB_RANK, and MaxJobRetirementTime.

Other Software Requirements for Rendering

For a complete open source solution, each machine acting as an Executer must have Blender installed on it. However, for rendering 3ds Max files, it would be possible to install 3ds Max Backburner, etc., on each Executer.

- Python on submitting machine
- WxPython needs to be installed

Grid Render setup

It is very simple after all this installation the grid-render needs to be copied into python folder on grid job submission machine. Once it is copied it can be executed by simple python command line statement

C:\Python25>python.exe Pro1-0.py

Appendix – C

Setting up Digital Puppetry

	eMove Server		eMove Client
1.	Start Hamachi and ask your client to connect Hamachi, check for his status in Hamachi when he is connected proceed to following instructions		
		2.	Connect Hamachi and check for the server if it is connected. You can ping each other from command prompt to see if connected.
3.	• Start Display.exe • Check the suit type, (should be torso head, in case of eMove suit) • Enable the server		
4	Start eMove-chat, Ask other party to start eMove-chat client		
		5	Start eMove-chat client
		6	Check the IP and Connect (check the sound if any party does not listen anything check your headphone and microphone settings on both sides)
7	Accept the call (check the sound if any party does not hear anything then check headphone and microphone settings on both sides)		
		8	Start MotionBuilder
		9	Drag IGS Slave
		10	Set the connect IPs inside MotionBuiler to Hamachi IPs and then Connect
		11	Inside the MotionBuilder, Menus >

			Create Binding > Online > Live > Recording
		12	Record > overwrite > Play x 2
		13	Live and recording off > First frame
		14	Left arm rotate +75 > key
		15	Right arm rotate -75 > key
		16	Drag character from asset browser to hips > Characterize > Biped
		17	Back to live in devices
		18	File Menu > Merge > Character
		19	Nav > Character > Character Name > Input Type to Character > Active
		20	Nav > Devices > Voice > Source to sound card in use (e.g. Sigma tell in our case) > click 'Live'
		21	Alt + Enter
		22	Inside eMove-chat start webcam for server to see clients • eMove-chat menus Chat > click 'Voice chat' • Type server IP and call
23	Inside eMove-chat accept the incoming call.		

Appendix – D

Snapshots

Figure. D-1: HyperFighter (first real-time motion capture) game

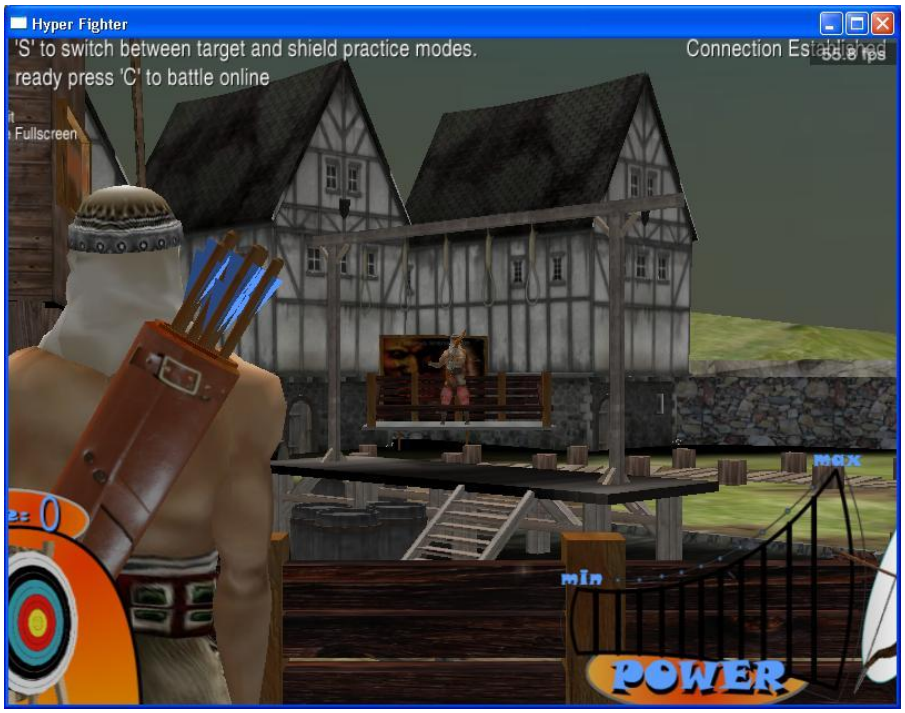

Figure. D-2: HyperFighter (first real-time motion capture) game

Figure. D-3: Two developers of HyperFighter game in motion capture suits

Figure. D-4: eMove suit early illustration and new final CAD design

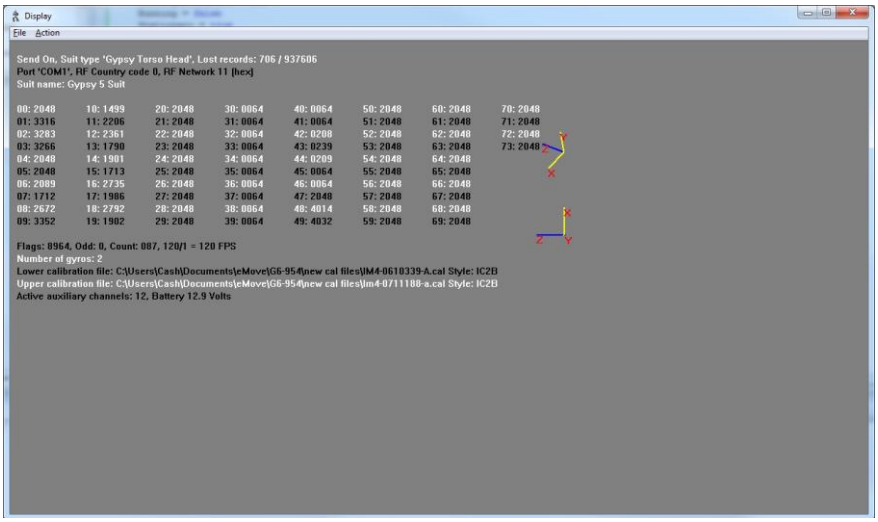

Figure. D-5: display.exe file connected eMove suit

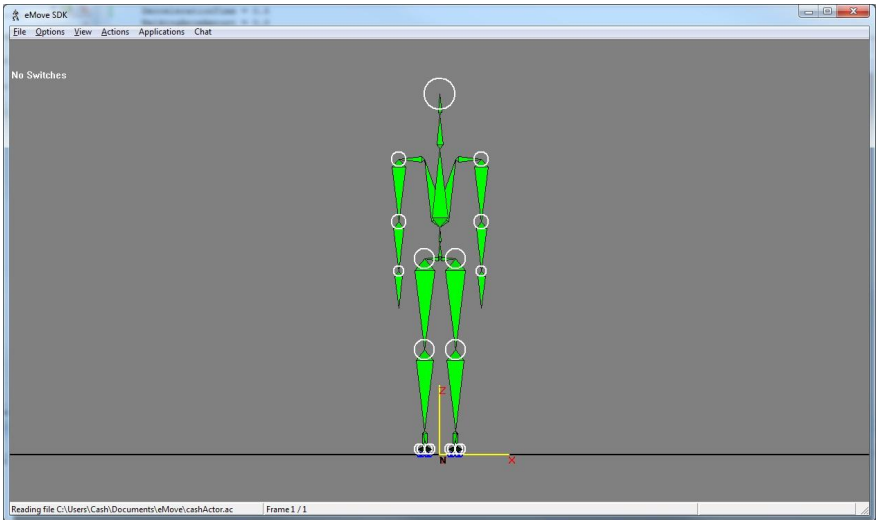

Figure. D-6: eMove-SDK connected with display.exe

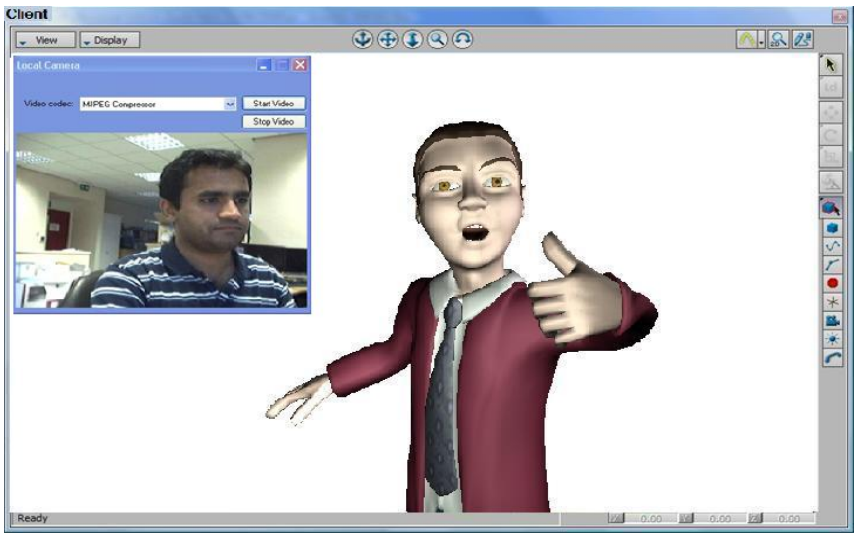

Figure. D-7: Digital puppetry application snapshot including eMove-chat

Figure. D-8: A rendered image from Fishbourne Roman Palace

Figure. D-9: A rendered image from Fishbourne Roman Palace

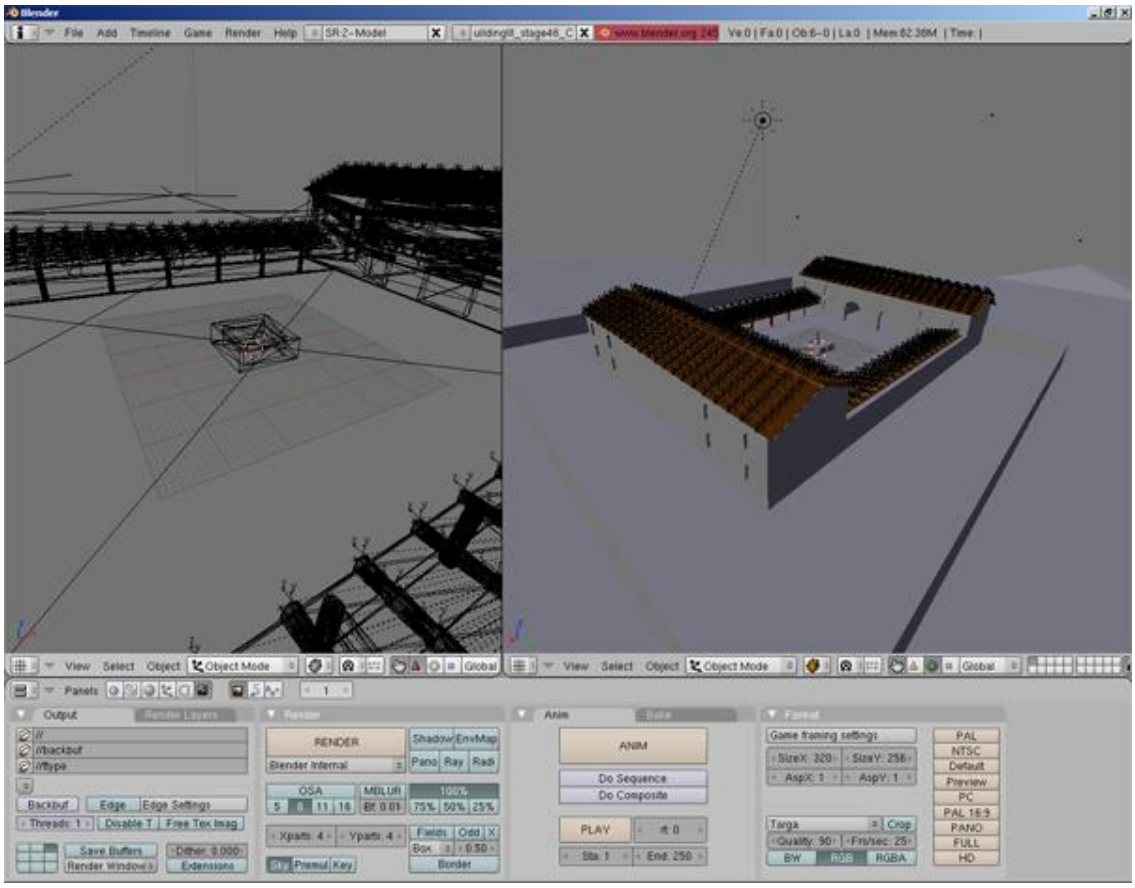

Figure. D-10: Fishbourne Roman Palace in Blender3D environment